|  |  |  |
| --- | --- | --- |
|  | ***Федеральное агентство по рыболовству***  ***Федеральное государственное бюджетное образовательное***  ***учреждение высшего образования***  ***«******Астраханский государственный технический университет»***  Система менеджмента качества в области образования, воспитания, науки и инноваций сертифицирована DQS  по международному стандарту ISO 9001:2015 | |
| Институт Информационных технологий и коммуникаций  Направление  09.03.01 Информатика и вычислительная техника  Профиль Автоматизированные системы обработки информации и управления  КафедраАвтоматизированные системы обработки информации и управления  от \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 № \_\_\_\_\_\_  **ОТЧЕТ ПО ПРАКТИКЕ**  Вид практики: эксплуатационная  Место прохождения практики: ***Астраханский государственный технический университет,  кафедра «Автоматизация сопровождение образовательного процесса в организации Региональный школьный технопарк»*** | | |
|  | | Отчет выполнил:  обучающийся группы ДИНРб-41  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кузургалиев Р.А. |
|  | | Руководитель практики от Университета  доктор технических наук, профессор  \_\_\_\_\_\_\_\_\_\_\_\_ Хоменко Т.В.  Результаты защиты отчета  Оценка, полученная на защите  «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»  Члены комиссии:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Шуршев В.Ф.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Хоменко Т.В.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г. |
| **Астрахань 2025** | | |

**ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ**

**Вид практики**: эксплуатационная

**Тип практики:**проектно-технологическая

**Способ проведения практики:** стационарная

**Обучающийся**   *Кузургалиев Радмир Алексеевич*

**Курс   4**     **группа**  *ДИНРб-41*

**Направление** 09.03.01 Информатика и вычислительная техника

**Профиль**  «Автоматизированные системы обработки информации и управления»  
**Место проведения практики** *Астраханский государственный технический университет,   
кафедра «Автоматизированные системы обработки информации и управления»*

**Объем и краткое содержание (виды работ) практики**

| **№**  **п/п** | **Раздел практики** | **Формы текущего контроля успеваемости** |
| --- | --- | --- |
| **1** | **1 этап (подготовительный)**  1. Получение индивидуального задания на практику.  2. Инструктаж по технике безопасности при работе на персональном компьютере и в компьютерном классе | Регистрация в журнале по технике безопасности, собеседование |
| **2** | **2 этап (теоретический )**  **Выполнение индивидуального задания: анализ предметной области**  1. Проведение анализа предметной области в соответствии с утвержденным планом;  2. Поиск и анализ информации по индивидуальному заданию; сбор данных, необходимых для решения поставленных задач. | Техническое задание.  Разделы отчёта по практике «Технический проект». |
| **3** | **3 этап: подготовка отчета по практике**  Подготовка отчета, в котором должны быть отражены результаты работы.  Составление пояснительной записки в виде файла и твердой копии.  Подготовка презентации для выступления на защите в комиссии кафедры | Материал по результатам работы: письмен­ный отчет, электрон­ная презентация |
| **4** | **Заключительный этап:**  Защита отчета по практике на кафедре | Отчет по результатам практики |
|  | **Форма отчетности по практике** | Зачет с оценкой |

Руководитель практики от Университета: *д.т.н., профессор Хоменко Татьяна Владимировна*

«27» января 2025 г. \_\_\_\_\_\_\_\_\_\_

*подпись*

Задание выдал: *д.т.н., профессор Хоменко Татьяна Владимировна*

«27» января 2025 г. \_\_\_\_\_\_\_\_\_\_

*подпись*

Задание получил: *Кузургалиев Радмир Алексеевич*

«27» января 2025 г. \_\_\_\_\_\_\_\_\_\_

*подпись*

**РАБОЧИЙ ГРАФИК (ПЛАН) ПРОВЕДЕНИЯ ПРАКТИКИ**

2024/2025 учебный год

**Направление**  09.03.01 Информатика и вычислительная техника

**Профиль**      Автоматизированные системы обработки информации и управления

**Курс**            **4** **группа**   *ДИНРб-41*  
**Место прохождения практики** Астраханский государственный технический университет,   
кафедра «Автоматизированные системы обработки информации и управления»

**Руководитель практики от Университета**  д.т.н., профессор Хоменко Татьяна Владимировна

**Вид практики:** эксплуатационная

**Тип практики:**проектно-технологическая

**Способ проведения практики**: стационарная

**Срок прохождения практики:** с 27.01.2025 по 11.02.2025

|  |  |
| --- | --- |
| **Дата (сроки)** | **Планируемые формы работы (раздел практики)** |
| 27.01.2025 | Знакомство с правилами внутреннего распорядка, инструктаж по технике безопасности.  Получение индивидуального задания на практику |
| 28.01.2025 - 01.02.2025 | Анализ предметной области, сбор информации, необходимой для разработки технического проекта и программы испытаний программного продукта. |
| 02.02.2025 | *Выходной день* |
| 03.01.2025 –  10.02.2025 | Подготовка отчета. Формирование пояснительной записки в виде файла и твердой копии. Подготовка презентации для выступления на защите. |
| 11.02.2025 | Защита практики |

Руководитель практики от университета: *д.т.н., профессор Хоменко Татьяна Владимировна*

«27» января 2025 г. \_\_\_\_\_\_\_\_\_\_

*подпись*

Обучающийся: *Кузургалиев Радмир Алексеевич*

«27» января 2025 г. \_\_\_\_\_\_\_\_\_\_

*подпись*

**ДНЕВНИК ПРАКТИКИ**

**Вид практики:** эксплуатационная

**Способ проведения практики:** стационарная

**Обучающийся** *Кузургалиев Радмир Алексеевич*, группа *ДИНРб-41*

**Направление**  09.03.01 Информатика и вычислительная техника  
**Профиль**         Автоматизированные системы обработки информации и управления

**Место проведения практики** *Астраханский государственный технический университет,   
кафедра «Автоматизированные системы обработки информации и управления»*

Дата начала практики: «27» января 2025 г.

Дата окончания практики: «11» февраля 2025 г.

| **Дата** | **Наименование  и ход работ,** | **Краткое описание работы** |
| --- | --- | --- |
| 27.01.2025 | Цели и задачи практики, техника безопасности в компьютерном классе, постановка задачи | Изучена инструкция по технике безопасности в компьютерном классе. Получено индивидуальное задание на практику. |
| 27.01.2025 -  30.01.2025 | Анализ предметной области. Сбор информации, необходимой для разработки технического проекта и программы испытаний программного продукта. | Собран и проанализирован материал по предметной области. |
| 31.01.2025 -  01.01.2025 | Анализ предметной области. Сбор информации, необходимой для разработки технического проекта и программы испытаний программного продукта. Разработка форматов данных, алгоритмов решения задачи. | Написано техническое задание. |
| 02.02.2025 | *Выходной день* |  |
| 03.02.2025 | Подготовка отчета. Составление пояснительной записки в виде файла и твердой копии. | Подготовлен отчет. Сформирована пояснительная записка. |
| 09.02.2025 |
| 10.02.2025 | Подготовка презентации для выступления на защите в комиссии кафедры | Подготовлена презентация для выступления на защите в комиссии кафедры. |
| 11.02.2025 | Защита практики |  |

Руководитель практики от университета: *д.т.н., профессор Хоменко Татьяна Владимировна*

«11» февраля 2025 г.\_\_\_\_\_\_\_\_\_\_

*подпись*

Обучающийся: *Кузургалиев Радмир Алексеевич*

«11» февраля 2025 г. \_\_\_\_\_\_\_\_\_\_

*подпись*

**Содержание**

[**ВВЕДЕНИЕ** 7](#_Toc189513132)

[**1** **ТЕХНИЧЕСКИЙ ПРОЕКТ** 8](#_Toc189513133)

[**1.1** **Предметная область** 8](#_Toc189513134)

[**1.1.1** **Общая информация и организационная структура организации** 8](#_Toc189513135)

[**1.1.2** **Взаимодействие с другими организациями** 9](#_Toc189513136)

[**1.1.3** **Форма проведения образовательного процесса.** 10](#_Toc189513137)

[**1.1.4** **Нормативное регулирование деятельности организации** 10](#_Toc189513138)

[**1.1.5** **Инфологическая модель предметной области.** 12](#_Toc189513139)

[**1.1.6** **Бизнес-процессы проблемной зоны предприятия** 15](#_Toc189513140)

[**1.1.7** **Описание информационных объектов.** 18](#_Toc189513141)

[**1.1.8** **Эскизный проект.** 19](#_Toc189513142)

[**1.1.9** **Спецификация требований к проектному решению.** 22](#_Toc189513143)

[**1.1.10** **Обзор аналогичных решений.** 24](#_Toc189513144)

[**1.1.10.1** **ФГИС «Моя школа»** 24](#_Toc189513145)

[**1.1.10.2** **«Сферум»** 25](#_Toc189513146)

[**1.1.10.3** **«Google Класс»** 25](#_Toc189513147)

[**1.1.10.4** **Сравнительная таблица аналогов** 25](#_Toc189513148)

[**1.1.10.5** **Достоинства и недостатки аналогов** 26](#_Toc189513149)

[**1.2** **Цель создания автоматизированной (информационной) системы.** 27](#_Toc189513150)

[**1.3** **Технология обработки информации** 28](#_Toc189513151)

[**1.3.1** **Диаграмма вариантов использования** 28](#_Toc189513152)

[**1.3.2** **Алгоритм добавления входящей и исходящей документации** 28](#_Toc189513153)

[**1.3.3** **Алгоритм добавления резерва** 29](#_Toc189513154)

[**1.3.4** **Алгоритм добавления приказа об основной деятельности** 29](#_Toc189513155)

[**1.3.5** **Алгоритм присвоения номера документа в системе** 29](#_Toc189513156)

[**1.3.6** **Алгоритм добавления приказа о мероприятии** 29](#_Toc189513157)

[**1.3.7** **Алгоритм добавления акта об участии в мероприятии** 30](#_Toc189513158)

[**1.3.8** **Алгоритм добавления образовательного приказа.** 30](#_Toc189513159)

[**1.3.9** **Алгоритм добавления образовательной группы.** 30](#_Toc189513160)

[**1.3.10** **Алгоритм зачисления и отчисления ученика в учебную группу.** 31](#_Toc189513161)

[**1.3.11** **Алгоритм перевода ученика между группами.** 31](#_Toc189513162)

[**1.4** **Входные и выходные данные.** 31](#_Toc189513163)

[**1.5** **Прототипы пользовательских интерфейсов.** 34](#_Toc189513164)

[**1.6** **Требование к техническому и программному обеспечению.** 36](#_Toc189513165)

[**2** **ТЕХНИЧЕСКИЙ ПРОЕКТ** 37](#_Toc189513166)

[**2.1** **Общие сведения о работе системы.** 37](#_Toc189513167)

[**2.2** **Описание архитектуры системы** 37](#_Toc189513168)

[**2.3** **Инсталяция и выполнение программного продукта** 37](#_Toc189513169)

[**2.4** **Описание программы МОДЕЛИ:** 38](#_Toc189513170)

[**ЗАКЛЮЧЕНИЕ** 55](#_Toc189513171)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 56](#_Toc189513172)

[**ПРИЛОЖЕНИЕ 1** 57](#_Toc189513173)

[**ПРИЛОЖЕНИЕ 2** 61](#_Toc189513174)

[**ПРИЛОЖЕНИЕ 3** 64](#_Toc189513175)

[**ПРИЛОЖЕНИЕ 4** 66](#_Toc189513176)

[**ПРИЛОЖЕНИЕ 5** 67](#_Toc189513177)

**ВВЕДЕНИЕ**

«Региональный школьный технопарк» – это динамичное место, где активно развивается техническое и инженерное образование для школьников. Благодаря разнообразию образовательных программ и технических возможностей новых технологий, сотрудникам технопарка необходимо обладать инструментами для автоматизированного сопровождения образовательного процесса.

Робототехника, программирование и инженерное дело представляют собой области с высокой степенью индивидуализации, требующие внимательного контроля за каждой программой и ресурсом. Именно поэтому автоматизированный учет, а также возможность индивидуального подбора ресурсов, становятся важными аспектами для обеспечения высокого уровня образовательного процесса.

Центр образования создает потребность в эффективной логистике и управлении ресурсами. Информационная система «РШТ» будет способствовать упорядоченному протеканию процессов от поступления ресурсов до их использования в образовательных программах, что повысит качество предоставления образовательных услуг и сократит время реализации программ.

Важным аспектом является также анализ данных. Система предоставит возможность обработки документации, анализа эффективности учебных программ, определения популярности различных технических направлений и выявления динамики учебного процесса по времени. Эти отчеты будут служить ценным инструментом для принятия стратегических решений и утверждений образовательных программ.

Однако в данный момент организация эксплуатирует информационную систему, которая не способна полноценно сопровождать образовательный процесс.

Целью разработки информационной системы для организации "Региональный школьный технопарк" является создание программного продукта, сопровождающего образовательный процесс и документооборот в организации «Региональный школьный технопарк».

Назначение автоматизированной (информационной) системы «Регионального школьного технопарка» заключается в обеспечении бесперебойной работы системы автоматизации сопровождения и увеличении аналитических показателей образовательного процесса.

Таким образом, информационная система "Регионального школьного технопарка" не только улучшит оперативные процессы внутри образовательного учреждения, но и обеспечит ценную аналитику для успешного ведения образовательного процесса в условиях динамичного и конкурентного образовательного процесса.

1. **ТЕХНИЧЕСКИЙ ПРОЕКТ** 
   1. **Предметная область**
      1. **Общая информация и организационная структура организации**

"Региональный школьный технопарк" (РШТ) — это образовательная организация, подчиняющаяся Министерству образования Астраханской области и состоящая из нескольких ключевых структурных подразделений, каждое из которых выполняет уникальную роль в образовательном процессе. В состав «РШТ» входят:

1. Технопарк — основная площадка, оснащённая высокотехнологичным оборудованием, где реализуются инновационные образовательные программы.
2. Кванториум — подразделение, ориентированное на углублённое изучение современных технологий и инженерных дисциплин.
3. Центр одарённых детей (ЦОД) — направление, специализирующееся на работе с талантливыми детьми, предоставляющее им возможности для развития способностей.
4. Центр детского научно-технического творчества (ЦДНТТ) — площадка, где дети могут заниматься научно-техническим творчеством, развивая навыки проектной деятельности.

Миссия РШТ заключается в создании условий для погружения детей в мир инженерных профессий, развития их творческого потенциала и подготовки к будущей профессиональной деятельности в высокотехнологичных отраслях.

Стратегия организации основана на применении проектного подхода, который позволяет детям осваивать инженерные специальности через практическую работу над реальными задачами. РШТ оснащён современным оборудованием, что обеспечивает высокий уровень образовательного процесса и способствует формированию у учащихся навыков, востребованных в условиях быстроменяющегося мира.

В наше время, когда технологии играют ключевую роль в оптимизации бизнес-процессов, создание информационной системы для организации "Региональный школьный технопарк" представляет собой важный шаг вперёд для оптимизации образовательного процесса и повышения конкурентоспособности. Сфера образования постоянно эволюционирует, а требования становятся всё более индивидуализированными. В этом контексте актуализация учебного процесса через внедрение информационной системы обеспечивает несколько значимых преимуществ:

* Эффективное управление учебными проектами: Система позволяет автоматизировать учет учебных программ, упрощая их мониторинг и улучшая контроль над ресурсами и сроками выполнения.
* Индивидуальный подход к учащимся: Возможность создания персонализированных образовательных программ требует тщательного учета потребностей каждого учащегося. Информационная система обеспечивает точность в управлении такими программами.
* Аналитика и оценка эффективности: Система предоставляет возможность проводить анализ учебных программ и проектов, что является ключевым инструментом для выявления тенденций и принятия обоснованных стратегических решений.
* Оптимизация ресурсов и оборудования: Интегрированная система учета ресурсов и оборудования помогает эффективно управлять необходимыми ресурсами для проведения технических и инженерных проектов.
* Укрепление взаимодействия с партнерами: Сотрудничество с другими образовательными учреждениями облегчает обмен знаниями и опытом, способствует развитию образовательных программ и проектов.
* Улучшение обслуживания школьников: Автоматизация процессов позволяет обеспечивать более оперативную и точную информацию о мероприятиях технопарка, образовательных программах и доступе к ресурсам.
  + 1. **Взаимодействие с другими организациями**

"Региональный школьный технопарк" активно сотрудничает с такими крупными образовательными центрами "Сириус", что предоставляет ученикам «РШТ» уникальные возможности участия в различных мероприятиях и олимпиадах как самого "Сириуса", так и других образовательных платформ. Такое участие имеет несколько важных аспектов, которые стоит анализировать для "Регионального школьного технопарка".

Во-первых, анализ результатов участия учеников помогает оценить качество образовательного процесса и эффективность программ, разработанных технопарком. Это позволяет выявить сильные и слабые стороны в обучении, а также в подходах к подготовке учащихся к соревнованиям.

Во-вторых, участие в мероприятиях "Сириуса" предоставляет ученикам ценные практические навыки и опыт командной работы, что является критически важным для их личностного и профессионального развития. Это также способствует формированию положительного имиджа "Регионального школьного технопарка" и Астраханской области.

В-третьих, результаты и достижения учащихся на таких площадках служат основой для дальнейшего развития технопарка. Умение анализировать успехи и недостатки помогает в дальнейшем корректировать образовательные программы, разрабатывать новые курсы и мероприятия, что, в итоге, влияет на привлечение будущих студентов и партнеров.

* + 1. **Форма проведения образовательного процесса.**

Образовательный процесс в «Региональном школьном технопарке» проводится в учебных группах. Учебные группы — это небольшие коллективы учеников школьного возраста, собирающиеся для совместного обучения и обмена знаниями по предметам или проектам. Они необходимы для улучшения процесса обучения: обсуждение тем в группе помогает лучше усваивать материал, а поддержка участников способствует обмену ресурсами и информации. Кроме того, работа в команде развивает навыки сотрудничества и коммуникации, а также повышает мотивацию к учебе. Учебные группы также дают возможность услышать различные точки зрения, что помогает расширить горизонты знаний. Обычно в группе состоит 10-12 школьников, но в ходе деятельности их количество может измениться, так в ходе приёмной кампании обучающийся может перевестись, быть «дозачислен» в группу или вовсе отчислен.

Отличительной особенностью образовательного процесса «Регионального школьного технопарка» является то, что обучающийся может находится сразу в нескольких учебных группах одновременно.   
 Во время образовательного процесса важно отмечать явку учеников, так как это помогает контролировать их успеваемость и активность в обучении. Отслеживание явки позволяет преподавателям видеть, насколько активно участвуют студенты, что напрямую связано с их результатами. Регулярная явка способствует формированию ответственности у студентов и создает атмосферу обязательности. Собранные данные о явке могут помочь в анализе учебного процесса и выявлении слабых мест, что позволяет корректировать подход к обучению. Также отметка явки может выступать дополнительным стимулом для студентов, поскольку осознание того, что их участие фиксируется, может повысить мотивацию посещать занятия и активно участвовать в процессе. Таким образом, отметка явки является важным инструментом для обеспечения качества образования и создания комфортной учебной среды.

* + 1. **Нормативное регулирование деятельности организации**

В ходе работы «Регионального школьного технопарка» необходимо активно взаимодействовать с разнообразной документацией, включая входящие и исходящие документы и приказы. Документооборот играет важную роль в современных организациях. Он позволяет обеспечить прозрачность бизнес-процессов, сохраняя следы действий и решений, а также разграничивать доступ к конфиденциальным данным. Кроме того, правильно настроенный документооборот способствует соблюдению законодательных требований, повышает эффективность бизнес-процессов, упрощает аудит и анализ, а также улучшает безопасность обработки информации.

Входящая документация представляет собой информацию, поступающую в организацию от внешних источников, таких как письма, факсы, электронные письма, отчеты, заявки и другие документы от партнеров, клиентов, поставщиков и государственных органов. Она служит основой для принятия решений, выполнения задач и обеспечения необходимой информацией для деятельности технопарка.

Исходящая документация, напротив, создается и отправляется из организации. Это могут быть договора, отчеты, письма, коммерческие предложения, инструкции, уведомления и другие документы, которые представляют интерес для сторонних организаций, клиентов, партнеров или государственных учреждений.

Для организации важно иметь возможность отслеживать информацию о том, кто загрузил файлы в систему, кто ответственен за исполнение приказов и постановлений, а также кто участвует в процессе подписи документов. Это позволит установить прозрачность и ответственность в рамках рабочих процессов, обеспечивая контроль за ходом выполнения задач и управление доступом к информации. Такая функциональность поможет повысить качество работы и сделать взаимодействие между участниками процесса более удобным и прозрачным.  
 Кроме того, важно учитывать, что существуют определенные типы документов, на которые необходимо отвечать в официальной форме. Это подчеркивает важность точного выполнения требований по оформлению документов и обеспечивает правильное ведение деловой переписки. Учитывая этот аспект, система управления документами должна предоставлять возможность определения и отслеживания обязательных форматов ответов на различные типы запросов или документов, обеспечивая соответствие установленным нормам и процедурам.

Особой разновидностью документов в «Региональном школьном технопарке» являются приказы. Выделяются три типа приказов:

* Приказы об основной деятельности носят исключительно административную роль. Они могут касаться структуры управления, назначения ответственных лиц, утверждения графиков работы и распределения ресурсов. Основная цель — обеспечить функционирование учреждения, поддерживая порядок и дисциплину в административных аспектах. Также могут отменять действие других приказов.
* Приказы об образовательной деятельности определяют образовательную деятельность организации. Они регулируют взаимоотношение между учебными группами и учениками, включают в себя установление учебных планов, формат проведения занятий и контроль за успеваемостью учеников. На основе приказов этого типа производится зачисление, отчисление и перевод обучающихся. Главная задача — создать оптимальные условия для образовательного процесса, способствующие развитию учащихся и поддерживающие их взаимодействие с педагогами.
* Приказы об участии в мероприятиях определяют участие технопарка и его учеников в различных научных, культурных и спортивных мероприятиях. Они могут касаться как внутренней организации мероприятий (конкурсов, выставок, олимпиад), так и внешнего сотрудничества с другими учреждениями. Ученики могут участвовать индивидуально, так и в составе команд в различных номинациях. Основная цель — активное участие учеников в разнообразных событиях для повышения их навыков, расширения кругозора и развития социальных связей.
  + 1. **Инфологическая модель предметной области.**

В ходе анализа предметной области были разработаны диаграммы классов. Они представлены в Приложении 2 на рисунках П2.1 – П2.5.

В классе *Должность* представлены все должности, которые могут занимать сотрудники «РШТ».

Класс *Пользователь* отражает всю информацию о человеке, как о сотруднике «Регионального школьного технопарка»:

* Информация необходимая для входа в систему (логин и пароль).
* Электронная почта.
* Должность пользователя.

Класс *Человек* представляет собою информацию о физическом лице, которое взаимодействует с «РШТ». К ним относятся сотрудники, ученики, представители власти и компаний-партнёров. Сущность содержит следующую информацию:

* Персональная информация о человеке (ФИО, дата рождения, пол).
* Организация, в которой осуществляет деятельность физическое лицо (в основном «Региональный школьный технопарк»).
* Отдел «РШТ».

Класс *Входящая документация* является центральной сущностью модуля работы с входящей документацией и включает в себя следующую информацию:

* Дата получения документа.
* Номер документа внутри организации.
* Способ получения.
* Краткая информация о документе (ключевые слова, тема документа).
* Необходимость ответа на документ (например, если это был запрос), а также крайняя дата ответа.

Класс *Исходящая документация* является центральной сущностью модуля работы с исходящей документацией и включает в себя следующую информацию:

* Дата получения документа.
* Номер документа внутри организации.
* Способ отправки.
* Краткая информация о документе (ключевые слова, тема документа).
* Ссылка на входящий документ, на который был выполнен ответ.

Класс *Приказ* является центральной сущностью, которое регулирует нормативно-правовые отношения внутри «Регионального школьного технопарка». Следующая информация является общей, вне зависимости от типа приказа:

* Дата приказа.
* Внутренний номер.
* Наименование.
* Тип приказа (об основной и образовательной деятельности, о мероприятиях)
* Ссылки на сотрудников «РШТ» (ответственные за исполнение, педагоги, вносящие приказ на обсуждение, подписывающий приказ).
* Краткая информация о документе (ключевые слова).

Класс *Положение* содержит основную информацию об административной деятельности организации.

Класс *Мероприятие* можно разделить на следующие подклассы: внутренние и внешние.

Класс *Внутреннее мероприятие* является представлением конкурсов, проектных защит, которые проводятся в рамках образовательного процесса внутри «РШТ». Сущности характерны следующие атрибуты:

* Название мероприятия.
* Уровень мероприятия.
* Временные рамки проведения мероприятия.
* Ответственные за проведение мероприятия.
* Тематика мероприятия.

Класс *Внешнее мероприятие* является представлением конкурсов, проектных защит, олимпиад, которые проводятся вне стен организации. Классу свойственны следующие атрибуты:

* Название мероприятия.
* Организатор.
* Место и временные рамки проведения мероприятия.
* Возрастные рамки участников.
* Краткая информация о мероприятии (ключевые слова).

Класс *Файл* является абстракцией файла в привычном понимании этого слова. Данная сущность содержит следующие данные:

* Тип файла.
* Расположение файла.
* Назначение файла.

Класс *Акт участия* является абстракцией факта участия обучающихся во внешних мероприятиях. Сами ученики выступают в индивидуальном или в командном формате. Во втором случае в представлении имени команды помогает класс *Команда*. Сущность Акты участии содержит следующую информацию:

* Ссылка на участников мероприятия.
* Ссылка на мероприятие.
* Ссылка на педагогов, сопровождающих обучающихся на мероприятиях.
* Направленность мероприятия.
* Формат проведения мероприятия
* Ссылка на команду.
* Номинация.

Класс *Учебная группа* представляет информацию об учебной группе в которых проводится обучение. Для данной сущности характерны следующие атрибуты:

* Номер группы.
* Ссылка на учебный план.
* Ссылка на педагога, который проводит занятия.
* Дата начала и окончания занятий.

Класс *Ученик* представляет информацию об обучающихся в рамках учебных групп:

* Количество баллов, набранных по ходу обучения в группе.
* Статус обучения в группе (зачислен, переведен в группу/из группы , отчислен, завершил обучение).
* Ссылка на учебную группу.
* Ссылка на сертификат, полученный по окончании обучения.

Выделим такой тип классов, как словарь. Словарь – это класс, основное назначение которого заключается в хранении дискретных значений и методов их обработки и интерпретации.

Словарь *Номенклатура* содержит всю необходимую информацию о видах приказов и их кодовых обозначениях в организации.

Словарь *Форма представления* содержит информацию о виде проведения мероприятий.

Словарь *Отдел* описывает структурные подразделения. Основные отделы были описаны в предметной области.

Словарь *Уровень мероприятия* представляет уровень проведения мероприятий.

Словарь *Направление мероприятия* олицетворяет направление, по которому проводится мероприятие.

Словарь *Тип мероприятия* обозначает какой характер носит проводимое мероприятие.

Функционирование программы для информационной системы «Регионального школьного технопарка» может быть сложным и включать различные алгоритмы для обработки запросов, кэширования, анализа данных и многих других задач. Ниже приведены общие этапы, которые включены в работу системы.

* + 1. **Бизнес-процессы проблемной зоны предприятия**

Можно выделить несколько основных процессов, которые сопровождаются потоками входных и выходных данных:

* 1. Обработка документации.
  2. Обработка приказов.

Процессы «Обработка документации» и «Обработка приказов» можно детализировать на подпроцессы. Таким образом процесс «Обработка документации» отображен на рисунке П3.1 в Приложении 3, разбивается на подпроцессы:

1. Получение «бумажного» варианта.
2. Заполнение данных о документе.
3. Прикрепление скан-копии документа.
4. Прикрепление «файла-ответа».
5. Сохранение документации.

Процесс «Обработка приказов» можно разбить на 3 подпроцесса:

1. Приказы об основной деятельности
2. Приказы об участии в мероприятиях.
3. Приказы об образовательной деятельности.

Диаграмма подпроцесса «Приказы об участии в мероприятиях» представлена на рисунке П3.2 в приложении 3, в которой можно выделить следующие подпроцессы:

1. Заполнение данных о приказе.
2. Создание карточки о мероприятии.
3. Создание команд и номинаций.
4. Назначение ответственных.
5. Прикрепление скан-копии документа.
6. Сохранение в системе.

Диаграмма подпроцесса «Приказы об образовательной деятельности» представлена на рисунке П3.3 в приложении 3, в которой можно выделить следующие подпроцессы:

1. Заполнение данных о приказе.
2. Выбор отдела и типа приказа.
3. Выбор учебной группы.
4. Выбор учащегося.
5. Назначение ответственных.
6. Прикрепление скан-копии документа.
7. Сохранение в системе.

Диаграмма подпроцесса «Приказы об основной деятельности» представлена на рисунке П3.4 в приложении 3, в которой можно выделить следующие подпроцессы:

1. Заполнение данных о приказе.
2. Назначение ответственных.
3. Прикрепление другого приказа/положения.
4. Выбор действия (отмена/изменение приказа/положения).
5. Прикрепление скан-копии документа.
6. Сохранение в системе.

Как было проанализировано в предметной области в рассматриваемой системе существуют следующие сущности: *Документация* и *Ученик*. В Приложении 4представлены диаграммы состояний для этих сущностей.

Изначально в системе документ отсутствует, при его создании с помощью метода **Create** пребывает в активном состоянии, после чего на него может быть указан ответ с помощью метода **Update,** приобретая состояние «Указан ответ» или быть сразу сохраненным в системе методом **save,** тем самым переходя в состояние «Сохранён», также есть возможно изменения статуса на «Требуется указать ответ» с помощью **Create**. Документ может быть удалён с помощью метода **Delete.**

Класс Ученик находится изначально в неопределённом состоянии, затем переходит после появления приказа о зачислении в состояние «Зачислен» с помощью метода **Enroll**. После этого ученика можно с помощью приказа об образовательной деятельности перевести в другую группу, тем самым обучающийся находится в состояние «Переведён» благодаря методу **Transfer**. У ученика, находящегося в группе, есть два конечных состояния «Отчислен» и «Завершил обучение» (методы **Deduct** и **Graduate** соответственно)

«Региональный школьный технопарк» в рамках образовательного процесса в течение учебного года проводит приёмные кампании и летние смены для школьников. Это приводит к созданию огромного объёма данных, которые необходимо хранить и обрабатывать. Так со второй половины 2018 года по настоящий момент в «РШТ»:

* было подано более 10000 заявок;
* прошли обучение более 8000 учеников;
* обучающиеся были представлены на 435 мероприятиях по всей России;
* был получен 3861 сертификат.

За эти годы были сформированы:

* 1361 учебная группа;
* 403 конкурсная команда.

К каждой группе необходимо прикрепить преподавателя, авторскую программу, составить расписание и связать это с приказами об образовательной деятельности и мероприятиях. Это необходимо для регламентирования деятельности организации. Становится очевидным, что такой объём информации невозможно обработать вручную, к тому же становится актуальной проблема сопровождения образовательного процесса.

«Региональный школьный технопарк» уже имеет информационную систему, но она обладает несколькими недостатками:

**Приказы:**

* + Существующая система не способна автоматически создавать учебную группу, прикреплять к ней учеников и учебный план, изменять статус обучающихся (зачислять, переводить их между группами и отчислять).
  + Существующая система не обрабатывает информацию о мероприятиях, которые проводит «РШТ» и другие организации, в которых участвуют обучающиеся.
  + В системе отсутствует нормативно-правовое сопровождение административного процесса через приказы по основной деятельности.

**База данных:**

* Не приведенная к третьей нормальной форме (3NF) база данных может привести к избыточности данных, сложности в обновлении и удалении информации, а также повышенному риску ошибок и противоречий. Это усложняет поддержку и расширение системы, а также может привести к увеличению времени выполнения запросов к базе данных.
* Отсутствие у запросов к базе данных свойства целостности: при непредвиденных ошибках, таких как некачественное интернет-соединение, запрос может выполнится неполностью, что может привести к потере данных.

**Проблемы при масштабировании системы:**

* Наличие проблем при масштабировании может означать, что система не способна обрабатывать увеличенное количество пользователей, данных или транзакций без серьезных изменений в ее архитектуре. Это может привести к снижению производительности, недоступности системы в периоды пиковой нагрузки или дополнительным затратам на обновление аппаратного обеспечения.
* Система разработана не в рамках архитектуры (MVC + Service & Repository), что усложняет обслуживание системы для разработчика.

**Проблемы с обеспечением доступа к файлам:**

* Возникающие проблемы с обеспечением доступа к файлам могут затруднять управление, безопасность и обмен информацией, которую система учитывает. Это может приводить к потере данных, конфликтам при обновлении файлов, а также негативно сказываться на работе пользователей.
  + 1. **Описание информационных объектов.**

Как было сказано ранее «Региональный школьный технопарк» работает с различными формами документации, как в электронном, так и в бумажном виде. Особенно усложняет работу сотрудничество с другими организациями и ведомствами, так как у них структура работы зачастую построена по-другому. Для упрощения работы организации крайне важно иметь обладать сразу двумя этими формами (то есть скан-копией и редактируемой формой документа). На рисунке 1.1 приведен пример документа, с которым работает «РШТ».

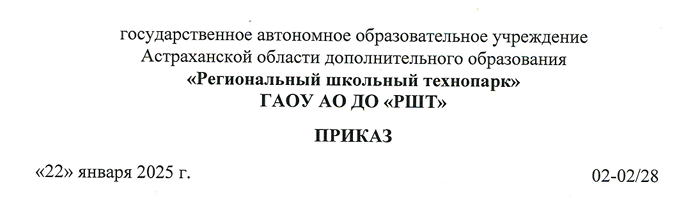


Рисунок 1.1 - Формат приказа

Как было сказано раннее необходимо определять людей, связанных с работой образовательного процесса, так как это значительно повышает эффективность образовательного процесса и его аналитику. На рисунке 1.2 представлен шаблон документа (в данном случае приказа) и сотрудников, связанных с ним.

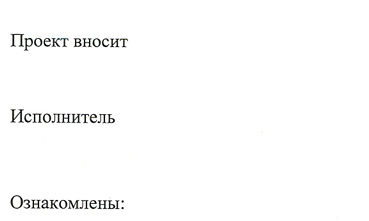


Рисунок 1.2 - Отражение связи вида «Приказ-Человек»

Если документ необходимо сформировать на основе имеющейся информации, то документ можно сгенерировать автоматически. На рисунке 1.3 продемонстрирован пример шаблон приказа об отчислении.

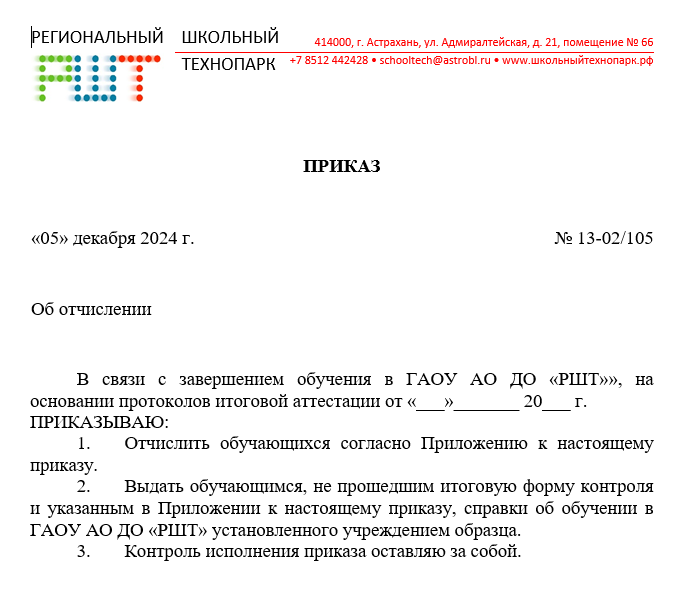


Рисунок 1.3 – Шаблон сгенерированного файла

* + 1. **Эскизный проект.**

В ходе анализа предметной области были рассмотрены следующие возможности реализации программного продукта:

* Desktop-приложение.
* Web-приложение.

Однако сразу выяснились основные минусы проектирования Desktop-приложения:

* Ограниченная доступность: Десктоп-приложения могут быть установлены только на определенные операционные системы (например, Windows, macOS, Linux), что ограничивает их использование для пользователей, которые предпочитают другие платформы.
* Требуется установка: для использования десктоп-приложений пользователи должны сначала установить их на свои компьютеры. Это может быть неудобно для пользователей, которые не хотят заниматься установкой программного обеспечения или имеют ограниченные права на установку на своих компьютерах.
* Обновления: Десктоп-приложения требуют регулярных обновлений для исправления ошибок и добавления новых функций. Пользователям приходится следить за новыми версиями и устанавливать обновления вручную.
* Зависимость от аппаратного обеспечения: Десктоп-приложения могут требовать определенных характеристик аппаратного обеспечения, таких как процессор, оперативная память или графический процессор. Это может ограничить использование приложения на старых или слабых компьютерах.
* Ограниченная мобильность: Десктоп-приложения не могут быть легко использованы на мобильных устройствах, таких как смартфоны или планшеты. Это может быть неудобно для пользователей, которые хотят иметь доступ к приложению в любое время и в любом месте.

В отличие от Desktop-приложения, у Web-приложения нет вышеописанных недостатков. Для корректной работы достаточно иметь браузер и интернет-соединение.

Следом необходимо выбрать архитектуру разрабатываемой системы. Фреймворк - это набор библиотек, инструментов и структур, предназначенных для упрощения разработки программного обеспечения. Для создания программного продукта целесообразно использовать фреймворк Yii. Он предоставляет разработчикам готовые компоненты, шаблоны и архитектурные принципы, которые позволяют создавать приложения более эффективно и структурированно.

Основные преимущества использования фреймворков включают:

* Ускорение разработки: Фреймворк предоставляет готовые компоненты и инструменты, которые позволяют разработчикам создавать приложения быстрее, поскольку им не нужно писать все с нуля.
* Структурирование кода: Фреймворк обычно предлагает определенную архитектуру приложения, что помогает разработчикам организовать свой код и следовать best practices.
* Улучшение безопасности: Многие фреймворки включают в себя встроенные механизмы безопасности, такие как защита от SQL-инъекций, CSRF и XSS атак.
* Поддержка множества технологий: Фреймворк может предоставлять интеграцию с различными технологиями, такими как базы данных, шаблонизаторы, аутентификация и многое другое.
* Сообщество и документация: Популярные фреймворки имеют большие сообщества разработчиков и обширную документацию, что облегчает поддержку и развитие проектов.

В рамках работы над данными необходимо определиться с тем, какую СУБД стоит использовать в рамках реализации программного продукта.

MariaDB — это система управления реляционными базами данных (СУБД), созданная как ответвление MySQL и также пользующаяся широкой популярностью среди разработчиков. Она предлагает надежное и высокоэффективное хранилище данных, подходящее для самых разнообразных приложений, от небольших веб-сайтов до крупных корпоративных систем. Использование MariaDB для разработки информационной системы для организации обосновано несколькими важными причинами:

* Открытый исходный код: MariaDB является проектом с открытым исходным кодом, что предоставляет разработчикам бесплатный доступ к СУБД и возможность модифицировать её под собственные нужды. Это особенно ценное преимущество для проектов с ограниченным бюджетом.
* Производительность и масштабируемость: MariaDB предлагает отличную производительность и возможность масштабирования. Это важно для организации "Региональный школьный технопарк", где база данных может обрабатывать большие объемы данных, такие как информация о заказах, комплектующих и клиентах.
* Совместимость с SQL: MariaDB придерживается стандартов SQL, что облегчает совместимость с другими базами данных. Это может оказаться полезным в будущем, если потребуется перенос приложения на другую платформу или использование дополнительных инструментов, поддерживающих SQL.
* Легкая интеграция: MariaDB хорошо интегрируется с различными языками программирования и технологиями, включая PHP. Эта гибкость позволяет разработчикам использовать разные технологии в различных частях системы.

Учитывая вышеописанные преимущества и недостатки, следует отметить, что в рамках реализации программного продукта функционала и возможностей СУБД MariaDB будет более чем достаточно.

Управление доступом — это критически важный аспект информационной безопасности, который защищает конфиденциальность данных и предотвращает несанкционированный доступ к личной и критически важной информации. Оно помогает снизить риски угроз, минимизируя вероятность утечек данных и инцидентов безопасности, контролируя, кто имеет доступ к определённым ресурсам.

RAC (Role-based Access Control) и PBAC (Policy-based Access Control) — это методы управления доступом, которые помогают контролировать, кто может получить доступ к определённым ресурсам в системах.

*RAC (Ролевое управление доступом):* здесь доступ к ресурсам определяется на основе ролей, назначенных пользователям. Каждая роль имеет свои права и привилегии. Это упрощает управление доступом, так как пользователям можно просто назначать роли вместо индивидуального назначения прав.

*PBAC (Политико-ориентированное управление доступом):* в этом случае доступ определяется на основе политик, которые могут учитывать различные факторы, такие как контекст (время, место, устройство и т. д.) и параметры пользователей. Это более гибкие и адаптивные механизмы, которые могут учитывать разнообразные условия.

Когда эти два подхода комбинируются (RAC + PBAC), создаётся более комплексная и адаптируемая система контроля доступа, которая может эффективно управлять правами пользователей с учётом разнообразных сценариев и условий. Это особенно полезно для сложных или крупных организаций, где требуются более строгие меры безопасности.

Хранение файлов в базе данных с использованием ссылок на файлы предполагает, что вместо загрузки самих файлов, таких как изображения или документы, в базу данных, сохраняются только ссылки или пути к этим файлам на файловой системе или в облачном хранилище. Это обеспечивает несколько преимуществ, включая повышенную производительность, так как базы данных обрабатывают меньше данных, что ускоряет выполнение запросов. Также упрощается управление файлами, поскольку их можно хранить на отдельных серверах, что облегчает резервное копирование, перемещение и удаление.

Дополнительное преимущество заключается в экономии места, поскольку в самой базе данных хранятся только метаданные и ссылки, а не большие файлы. Это позволяет гибко менять место хранения файлов или мигрировать на другое решение, не затрагивая записи в базе. Однако данный метод имеет и свои недостатки. Он может усложнить логику приложения, так как разработчику нужно координировать управление данными в базе и файлами на сервере или в облаке. Также существует риск несоответствия данных: если файлы были удалены или перемещены, ссылки в базе могут стать недействительными, приводя к ошибкам. Таким образом, «Региональный школьный технопарк» не только обеспечивает доступ к разнообразным образовательным программам, но и создает уникальный опыт для своих учащихся, включая индивидуальный подход, высокий стандарт образования и доступ к новейшим технологиям в области науки, техники и инженерии.

* + 1. **Спецификация требований к проектному решению.**

Заказчиком были определены требования к возможному решению вышеописанных проблем.

Выделим основные нефункциональные требования:

* Производительность и масштабируемость:
* Время отклика системы при переходах не должно составлять более трёх секунд.
* Система должна поддерживать одновременное подключение ста пользователей, то есть всех сотрудников «РШТ» и небольшой резерв для будущих сотрудников.
* Время выполнения операций также не должно составлять более трёх секунд.
* Возможность расширения инфраструктуры (например, добавление новых сервисов).
* Доступность:
* Система должна быть доступна в течение всего календарного года.
* Надежность и отказоустойчивость:
* Система должна продолжать работать корректно в случае сбоев.
* Каждый квартал производится резервное копирование всех данных системы.
* Аутентификация и авторизация пользователей.
* Удобство использования:
* Интерфейс должен быть интуитивно понятным для пользователей.
* Время на ознакомление с функционалом не должно составлять более двух рабочих недель.
* Поддержка различных устройств (мобильные устройства, ПК, ноутбуки).
* Поддержка и обслуживание:
* Простота обновления и модификации системы.
* Легкость в диагностике и исправлении ошибок.
* Документированность кода и архитектуры.

Информационная система должна отвечать следующим функциональным требованиям:

* Возможность добавления документации в систему.
* Открытый исходный код.
* Возможность создания учебных групп и прикрепления к ней учеников.
* Автоматическая привязка приказов об образовательной деятельности к образовательным программам и группам.
* Добавление информации о мероприятиях, проводимых в «РШТ» и за его пределами.
* Возможность добавления и удаления файлов из системы.
* Аналитика образовательного процесса.
* Возможность изменения статусов учеников в системе.

Также были определены основные требования по безопасности:

* защита системы должна обеспечиваться комплексом программно-технических средств и поддерживающих их организационных мер;
* доступ к внутреннему функционалу должен быть обеспечен только после аутентификации и авторизации пользователей;
* разграничение прав доступа пользователей и администраторов системы должно строиться по принципу «что не разрешено, то запрещено»;
* подключение клиентской части системы к серверной части должно осуществляться с использованием защищенных версий протоколов HTTPS;
* доступ к серверным файлам (файлы программного обеспечения и конфигурационные файлы) должен быть ограничен паролем администратора.
  + 1. **Обзор аналогичных решений.**
       1. **ФГИС «Моя школа»**

«Моя школа» — образовательная платформа для учеников, родителей и учителей. На ней можно получить доступ к цифровым сервисам и учебным материалам. Платформа создана Минпросвещения для реализации образовательных программ всех уровней основного образования.

Задачи платформы «Моя школа»:

* Создание современной и безопасной образовательной среды.
* Предоставление равного доступа к качественному образовательному контенту и цифровым сервисам по всей территории РФ для всех обучающихся.
* Повышение уровня цифровой грамотности педагогов с использованием дистанционных технологий.
* Создание возможностей для вовлечения родителей в процесс образования детей.
* Взаимодействие региональных и федеральных систем и использование единых классификаторов, реестров, справочников и форматов взаимодействия.

Согласно Приказу Минпросвещения России от 30.06.2021 г. № 396 «О создании федеральной государственной информационной системы Минпросвещения России «Моя школа», система создана, чтобы повысить уровень цифровой грамотности педагогов с использованием дистанционных образовательных технологий, создать современную и безопасную образовательную среду и возможности для вовлечения родителей (законных представителей) в процесс образования их детей.

Внедрение ФГИС призвано также создать условия для взаимодействия региональных и федеральных систем и использовать единые классификаторы, реестры, справочники и форматы взаимодействия, обеспечить равный доступ к качественному цифровому образовательному контенту и цифровым образовательным сервисам для всех категорий обучающихся.

* + - 1. **«Сферум»**

Сферум — это удобное и многофункциональное приложение, ориентированное на учителей, учеников и их родителей. Оно позволяет организовывать дистанционные уроки, что особенно актуально в современном образовательном процессе. Платформа предоставляет возможность отправлять домашние задания, информировать о важных событиях в школе и координировать взаимодействие между учителями и учениками. Сферум полностью бесплатен, что делает его доступным для всех пользователей.

Приложение функционирует как отдельное пространство внутри VK Мессенджера, что упрощает доступ к его функционалу. Внутри Сферум доступны каналы и группы, где можно найти всю необходимую информацию для учебы, от материалов до расписаний.

Используя Сферум, учителя могут провести видеоуроки, организовать опросы для проверки знаний, а ученики спокойно сдают свои домашние задания в установленный срок. Таким образом, Сферум становится не просто платформой для общения, а полноценным инструментом для образовательного процесса, который помогает сделать обучение более эффективным и организованным.

* + - 1. **«Google Класс»**

Google Класс — это обучающая платформа от Google, которая предназначена для упрощения процесса создания и управления учебными курсами и взаимодействия между преподавателями и учащимися. Она предоставляет инструменты для организации виртуального обучения, облегчая распределение материалов, оценку работ и обратную связь.

Одной из основных возможностей Google Класса является создание классов, где преподаватели могут добавлять студентов и делиться учебными материалами, такими как задания и ресурсы. Преподаватели могут создавать задания, задавать сроки и отслеживать выполнение работ, а также оставлять комментарии и выставлять оценки. Взаимодействие между учащимися и преподавателями становится более организованным благодаря разделам для обсуждений и возможностей комментирования.

Платформа также интегрируется с другими инструментами Google, такими как Google Диск, Google Документы и Google Таблицы, что позволяет легко делиться файлами и материалами.

Кроме того, Google Класс предоставляет возможность отслеживать успеваемость учащихся и анализировать их достижения, что помогает преподавателям адаптировать учебный процесс под нужды учащихся. Всё это делает Google Класс удобным инструментом для дистанционного и смешанного обучения.

* + - 1. **Сравнительная таблица аналогов**

В таблице 1.1 приведено сравнение аналогов по различным критериям.

Таблица 1.1 – Сравнение аналогов.

| **№** | **Характеристики** | **«Моя школа»** | **«Сферум»** | **Google Класс** |
| --- | --- | --- | --- | --- |
| 1 | Ориентированность на организации дополнительного образования | - | - | + |
| 2 | Открытый исходный код | - | - | - |
| 3 | Сопровождение учебных и внеучебных мероприятий | - | - | + |
| 4 | Учебные группы | + | + | + |
| 5 | Учёт достижений | + | + | + |
| 6 | Аналитика образовательного процесса | + | + | + |
| 7 | Сопровождение административного процесса | - | - | - |
| 8 | Возможность эксплуатации в государственной организации | + | + | - |
| 9 | Возможность работы с файлами | + | + | + |

* + - 1. **Достоинства и недостатки аналогов**

Одно из самых главных недостатка решения «Google Класс» – это его иностранное происхождение, что автоматически не позволяет его использовать в государственных учреждениях в связи с нормативно-правовыми актами РФ, регламентирующие импортозамещение программного обеспечения. Также недостатком является и то, что «Google класс» не способен вести учёт административного процесса, а это крайне важно для сопровождения образовательного процесса. В соответствии с данными двумя недостатками, «Региональный школьный технопарк» не может использовать и множество других иностранных программных продуктов, которые не обозревались в данном документе.

Решение «Моя школа» обладает крайне ограниченным функционалом с точки зрения сопровождения образовательного процесса, например, отсутствует возможность добавления учебных и внеучебных мероприятий. Связано это с тем, что информационная система «Моя школа» была разработана для организаций среднего и общего, а никак для центров дополнительного образования.

Программный продукт «Сферум» обладает рядом несомненных преимуществ таких, как бесплатная эксплуатация, возможность проведения видеоконференций, но имеет крайне ограниченные возможности с точки зрения аналитики индивидуальных достижений обучающихся, к тому же несмотря на достаточно обширное количество ролей, система обладает скудным функционалом администрирования, что усложняет эксплуатацию «Сферума» в организации.

* 1. **Цель создания автоматизированной (информационной) системы.**

Цель разработки информационной системы для организации "Региональный школьный технопарк" заключается в создании информационной системы, способной обеспечить сопровождение образовательного процесса и документооборота в организации «Региональный школьный технопарк».

Достижение данной цели сопровождается следующими задачами:

* проектирование базы данных;
* разработка программного продукта;
* тестирование полученной системы;
* внедрение информационной системы в организации.

В ходе работы будет проведен анализ существующей базы данных и системы, выявлены их основные недостатки, основные требования к хранению и обработке информации, а также разработана структура новой базы данных, учитывающая специфику организации "Региональный школьный технопарк" и обеспечивающая функционирование информационной системы.

Данная работа предполагает изучение основных принципов проектирования баз данных, использование соответствующих инструментов и технологий, а также практическую проверку разработанной модели на конкретных примерах.

Назначение автоматизированной (информационной) системы «Регионального школьного технопарка» заключается в обеспечении бесперебойной работы системы автоматизации образовательного процесса. Программный продукт позволяет автоматизировать образовательный и административный процесс в организации. В целом, система для «Регионального школьного технопарка» направлена на создание гибкой и адаптивной инфраструктуры, способной обеспечить успешное функционирование образовательной организации. Кроме того, создание удобной и организованной инфраструктуры для документооборота способствует налаживанию внутренней коммуникации и обмену информацией между преподавателями, администрацией и учащимися. Простота и быстрота доступа к нужной информации положительно сказываются на качестве образования и общей атмосфере в учреждении.

* 1. **Технология обработки информации**
     1. **Диаграмма вариантов использования**

Результатом анализа предметной области стала диаграмма вариантов использования, которая показана в Приложении 5 на рисунке П5.1. Исходя из диаграммы можно сделать вывод, что система содержит роли: «Преподаватель» и «Администратор». Вариант использования «Преподаватель» включает в себя следующие функции:

* Просмотр приказов по образовательной деятельности.
* Просмотр учебных планов.
* Получение уведомлений об ошибках в своих группах.
* Просмотр своих учебных групп.
* Редактирование своих учебных групп.

Вариант использования «Администратор» включает в себя следующие функции:

* Просмотр/редактирование/удаление приказов.
* Просмотр/редактирование/удаление документации.
* Просмотр/редактирование/удаление учебных групп и их участников.
* Просмотр/редактирование/удаление пользователей и их прав.
* Просмотр/редактирование/удаление учебных планов.
* Просмотр/редактирование/удаление мероприятий и актов участия.

Функционирование программы для информационной системы «Регионального школьного технопарка» может быть сложным и включать различные алгоритмы для обработки запросов, кэширования, анализа данных и многих других задач. Ниже приведены общие этапы и алгоритмы, которые включены в работу системы.

* + 1. **Алгоритм добавления входящей и исходящей документации**

1. Проверить права пользователя на проведение данной операции.
2. Получить от пользователя необходимую информацию о документе.
3. Если есть сканированная версия документа, то

прикрепить сканированные файлы к форме заполнения.

1. Если существует документ-ответ, то

указать документ-ответ.

1. Присвоить документу номер (см. алгоритм 1.7.6).
2. Создать SQL-запрос добавления записи в базу данных.
3. Произвести выполнение транзакции. Внести информацию в таблицы базы данных системы.
   * 1. **Алгоритм добавления резерва**
4. Проверить права пользователя на проведение данной операции.
5. Создать SQL-запрос добавления записи в базу данных.
6. Присвоить документу номер.
7. Произвести выполнение транзакции.
8. Внести информацию в таблицы базы данных системы.
   * 1. **Алгоритм добавления приказа об основной деятельности**
9. Проверить права пользователя на проведение данной операции.
10. Получить от пользователя основную информацию о документе.
11. Если есть сканированная версия документа, то

прикрепить сканированные файлы к форме заполнения.

1. Создать объекты класса OrderPeople для создания связи вида «приказ-ответственный за исполнение».
2. Присвоить документу номер (см. алгоритм 1.7.6).
3. Создать SQL-запрос добавления записи в базу данных.
4. Произвести выполнение транзакции.
5. Внести информацию в таблицы базы данных системы.
   * 1. **Алгоритм присвоения номера документа в системе**

**Дано**: объект класса DocumentIn / DocumentOut / OrderMain.

1. Произвести выборку среди документов всех типов по типу и текущему году.
2. Определить приказы, которые были внесены до и позже текущей даты.
3. Отсортировать приказы по дате, определить максимальные и минимальные номера.
4. На основе этих данных документу присвоить номер и постфикс, если он необходим.
5. Создать SQL-запрос добавления записи в базу данных.
6. Произвести выполнение транзакции.
7. Внести информацию в таблицы базы данных системы.
   * 1. **Алгоритм добавления приказа о мероприятии**
8. Проверить права пользователя на проведение данной операции.
9. Получить от пользователя основную информацию о приказе.
10. Если есть сканированная версия документа, то

прикрепить сканированные файлы к форме заполнения.

1. Внести информацию в форму о мероприятии и дополнительную информацию для генерации приказа.
2. При назначении ответственных создать объекты класса OrderPeople для обеспечения связи вида «приказ-ответственный за исполнение».
3. Заполнить акт(-ы) участия (см алгоритм 1.7.8).
4. Присвоить документу номер (см. алгоритм 1.7.6).
5. Создать SQL-запрос добавления записи в базу данных.
6. Произвести выполнение транзакции. Внести информацию в таблицы базы данных системы.
   * 1. **Алгоритм добавления акта об участии в мероприятии**
   1. Проверить права пользователя на проведение данной операции.
   2. Определить тип участия (индивидуальный/командный).
   3. Определить участников мероприятия.
   4. Получить от пользователя основную информацию о мероприятии.
   5. Если есть сканированная версия акта об участии, то

прикрепить сканированные файлы к форме заполнения.

* 1. Создать SQL-запрос добавления записи в базу данных
  2. Произвести выполнение транзакции.
  3. Внести информацию в таблицы базы данных системы.
     1. **Алгоритм добавления образовательного приказа.**

1. Проверить права пользователя на проведение данной операции.
2. Получить от пользователя основную информацию о документе.
3. Если есть сканированная версия документа, то

прикрепить сканированные файлы к форме заполнения.

1. Создать объекты класса OrderPeople для создания связи вида «приказ-ответственный за исполнение».
2. Определить тип образовательного приказа (о зачислении / переводе / отчислении) (см. алгоритмы 1.7.11 и 1.7.12).
3. Присвоить документу номер (см. алгоритм 1.7.6).
4. Создать SQL-запрос добавления записи в базу данных.
5. Произвести выполнение транзакции.
6. Внести информацию в таблицы базы данных системы.
   * 1. **Алгоритм добавления образовательной группы.**
7. Определить педагогический состав.
8. Определить образовательную программу.
9. Определить даты проведения занятий.
10. Присвоить группе номер.
11. Создать SQL-запрос добавления записи в базу данных.
12. Произвести выполнение транзакции.
13. Внести информацию в таблицы базы данных системы.
    * 1. **Алгоритм зачисления и отчисления ученика в учебную группу.**
14. Получить от пользователя тип образовательного приказа.
15. Если приказ об отчислении, то

определить группу отчисляемого.

выбрать отчисляемого из группы.

| определить группу зачисляемого

определить зачисляемого

1. Создать SQL-запрос добавления записи в базу данных.
2. Произвести выполнение транзакции.
3. Внести информацию в таблицы базы данных системы.
   * 1. **Алгоритм перевода ученика между группами.**
4. Определить группу откуда переводится обучающийся.
5. Определить группу куда переводится обучающийся.
6. Создать SQL-запрос добавления записи в базу данных.
7. Произвести выполнение транзакции.
8. Внести информацию в таблицы базы данных системы.

Алгоритмы в информационной системе организации "Региональный школьный технопарк" должны обеспечивать точность обработки данных, а также удовлетворять потребности пользователя.

* 1. **Входные и выходные данные.**

Рассмотрим основные форматы входных и выходных данных, представленные в системе. Валидация данных происходит автоматически на стороне клиента.

Входные данные:

В таблице 1.2 представлена форма авторизации.

Таблица 1.2 – Форма авторизации.

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип** | **Формат** |
| логин | строка | Согласно маске [\*@\*.\*](mailto:*@*.*) |
| пароль | строка | Требования к формату отсутствуют |

В таблице 1.3 представлены приказы в виде входных данных.

Таблица 1.3 – Приказы.

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип** | **Формат** |
| Дата приказа | дата | Стандартный формат ДД-ММ-ГГГГ |
| Наименование | строка | Только символы кириллицы, латиницы и цифры |
| Ключевые слова | строка | Требования к формату отсутствуют |
| Исполнитель | строка из выпадающего списка | Требования к формату отсутствуют |
| Инициатор | строка из выпадающего списка | Требования к формату отсутствуют |
| Ответственные | строка из выпадающего списка | Требования к формату отсутствуют |
| Номенклатура | строка из выпадающего списка | Требования к формату отсутствуют |
| Скан-копия приказа | файл | png, jpg, pdf, zip, rar, 7z, tag, txt |
| Приложения | файл | xls, xlsx, doc, docx, zip, rar, 7z, tag, txt |

В таблице 1.4 представлены входящая и исходящая документация.

Таблица 1.4 – Документация.

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип** | **Формат** |
| Дата документа | дата | Стандартный формат ДД-ММ-ГГГГ |
| Наименование | строка | Только символы кириллицы, латиницы и цифры |
| Ключевые слова | строка | Требования к формату отсутствуют |
| Получатель/отправитель | строка из выпадающего списка | Требования к формату отсутствуют |
| Скан-копия приказа | файл | png, jpg, pdf, zip, rar, 7z, tag, txt |
| Приложения | файл | xls, xlsx, doc, docx, zip, rar, 7z, tag, txt |

В таблице 1.5 представлены учебные группы.

Таблица 1.5 – Учебная группа.

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип** | **Формат** |
| Дата начала занятий | дата | Стандартный формат ДД-ММ-ГГГГ |
| Дата окончания занятий | дата | Стандартный формат ДД-ММ-ГГГГ |
| Отдел | строка из выпадающего списка | Требования к формату отсутствуют |
| Ключевые слова | строка | Требования к формату отсутствуют |
| Педагог | строка из выпадающего списка | Требования к формату отсутствуют |
| Образовательная программа | строка из выпадающего списка | Требования к формату отсутствуют |
| Обучающиеся | строка из выпадающего списка | Требования к формату отсутствуют |

Выходные данных продемонстрированы на примере следующих сущностей: приказы, документация и учебная группа. В таблице 1.6 представлен приказ.

Таблица 1.6 – Приказы.

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип** | **Формат** |
| Наименование | строка | Только символы кириллицы, латиницы и цифры |
| Ключевые слова | строка | Изменяемое поле ввода |
| Исполнитель | строка из выпадающего списка | Поле множественного выбора |
| Инициатор | строка из выпадающего списка | Поле множественного выбора |
| Ответственные | строка из выпадающего списка | Поле множественного выбора |
| Скан-копия приказа | файл | png, jpg, pdf, zip, rar, 7z, tag, txt |
| Приложения | файл | xls, xlsx, doc, docx, zip, rar, 7z, tag, txt |

В таблице 1.7 представлены входящая и исходящая документация.

Таблица 1.7 – Документация.

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип** | **Формат вывода** |
| Наименование | строка | Изменяемое поле ввода |
| Ключевые слова | строка | Изменяемое поле ввода |
| Получатель/отправитель | строка из выпадающего списка | Изменяемое поле ввода |
| Скан-копия приказа | файл | png, jpg, pdf, zip, rar, 7z, tag, txt |
| Приложения | файл | xls, xlsx, doc, docx, zip, rar, 7z, tag, txt |

В таблице 1.8 представлены учебные группы.

Таблица 1.8 – Учебная группа.

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип** | **Формат** |
| Ключевые слова | строка | Требования к формату отсутствуют |
| Педагог | строка из выпадающего списка | Поле множественного выбора |
| Образовательная программа | строка из выпадающего списка | Поле множественного выбора |
| Обучающиеся | строка из выпадающего списка | Поле множественного выбора |

* 1. **Прототипы пользовательских интерфейсов.**

В ходе анализа предметной области были определены следующие прототипы пользовательских интерфейсов:

Интерфейс index (представлен на рисунке 1.4), основная задача которого демонстрация всех сущностей одного типа в общем виде с основными характеристиками.

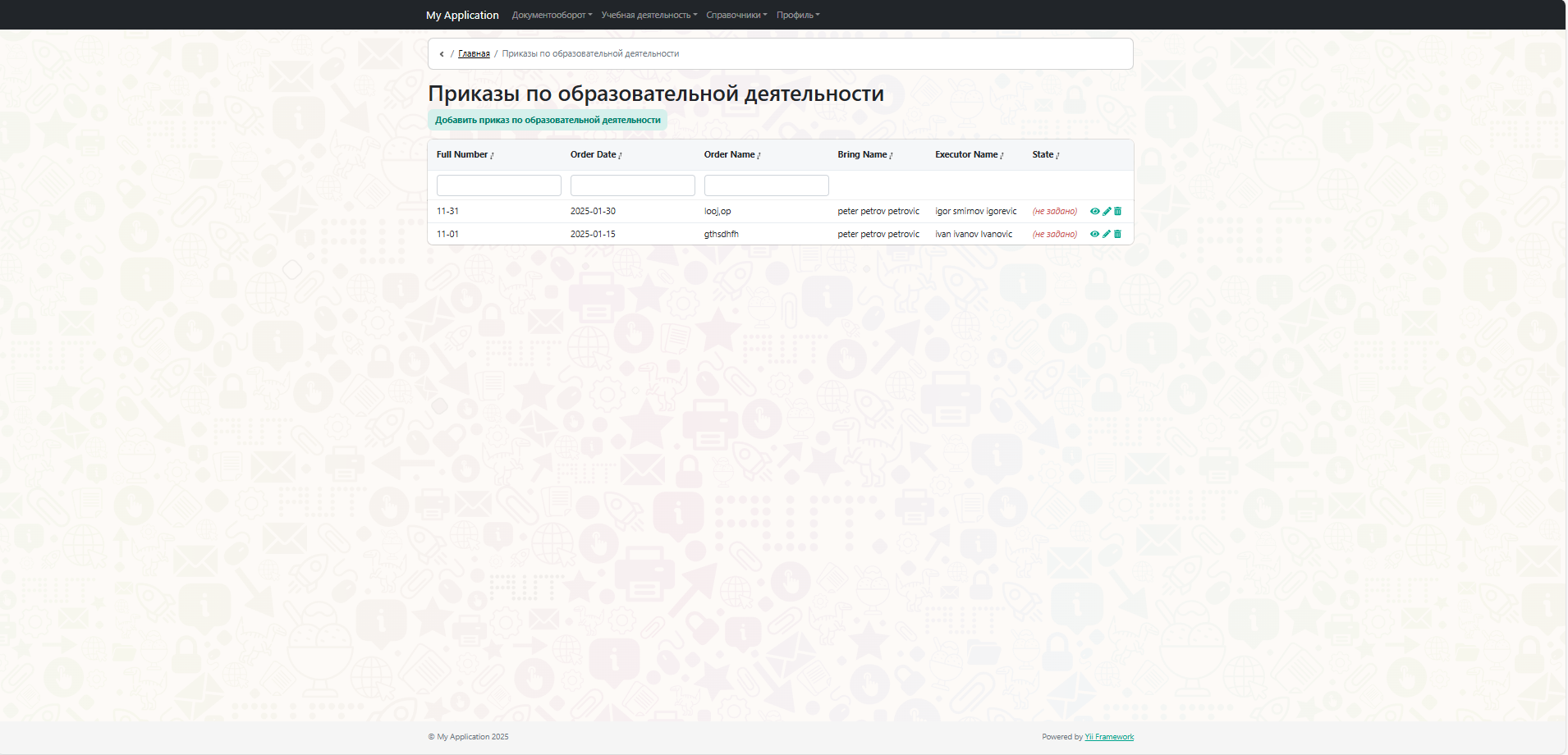


Рисунок 1.4 - Интерфейс index

Интерфейс create (представлен на рисунке 1.5), представляет собой форму ввода данных с различными полями: поля ввода, выпадающие списки, чекбоксы, динамические формы. С помощью интерфейса пользователь создаёт новые объекта класса.

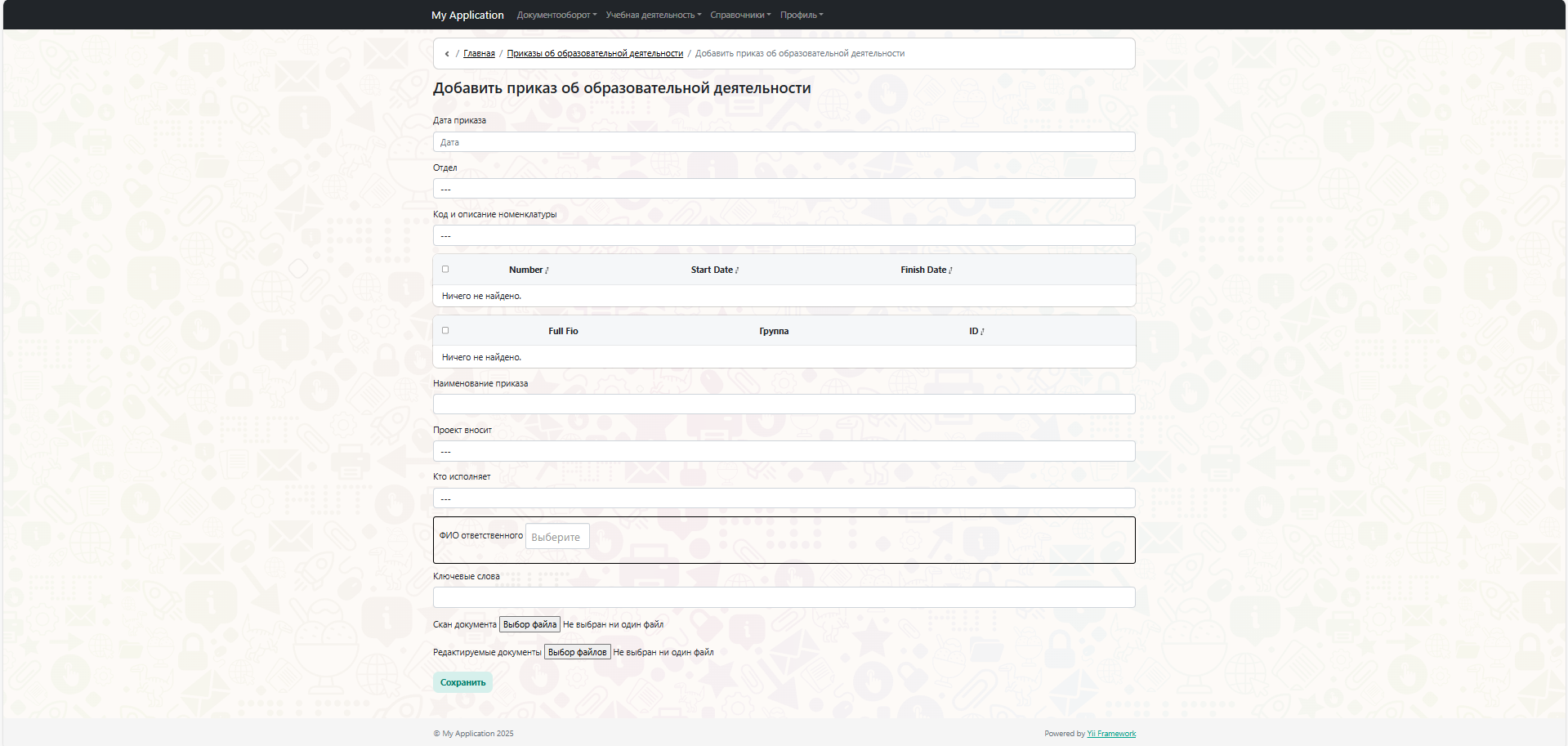


Рисунок 1.5 - Интерфейс create

Интерфейс update (представлен на рисунке 1.6), представляет видоизменённый интерфейс create, только с предзагруженными атрибутами. С помощью update пользователь изменяет элементы, созданные в интерфейсе create.

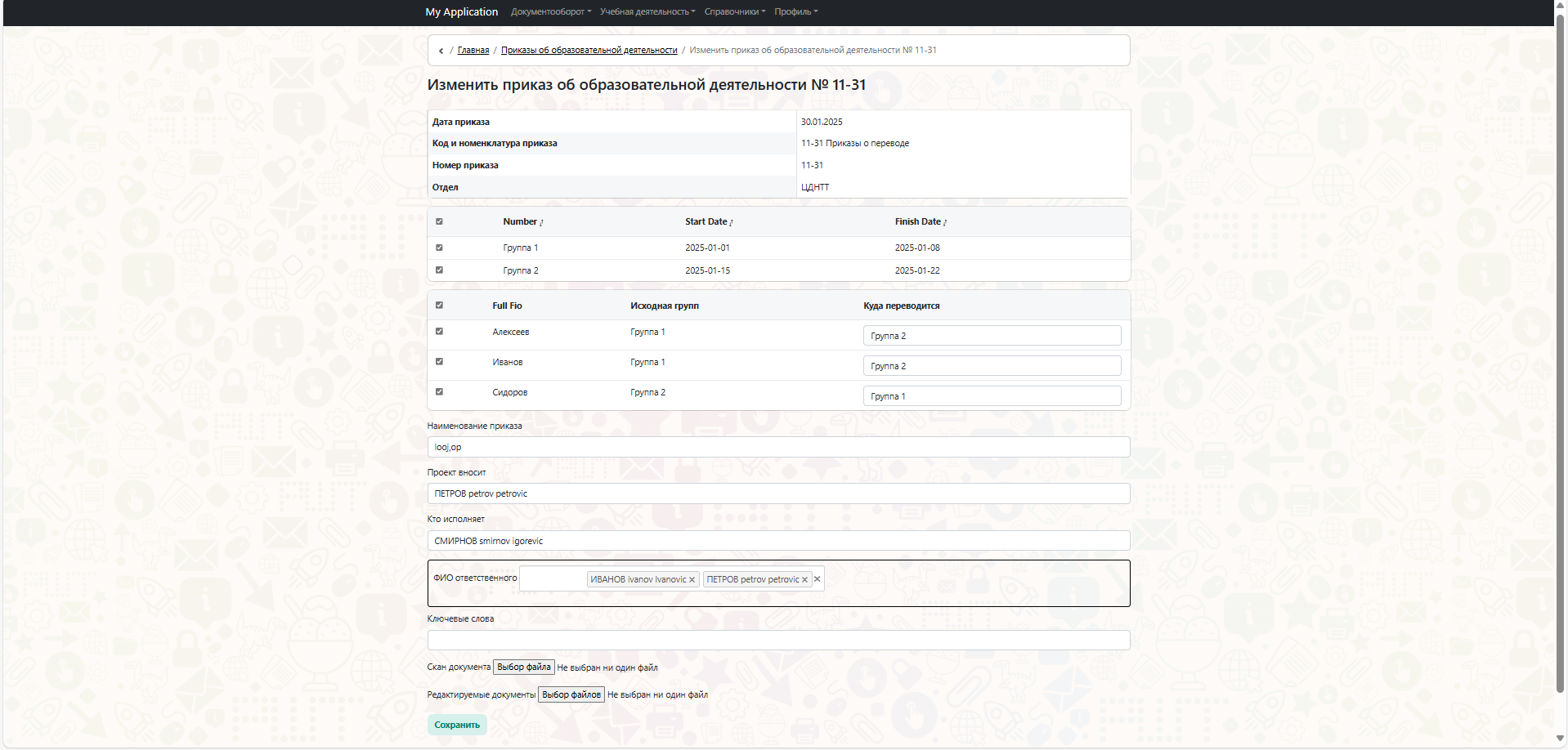


Рисунок 1.6 - Интерфейс update

Интерфейс view (представлен на рисунке 1.7), представляет собой интерфейс, который отображает информацию о сущности.

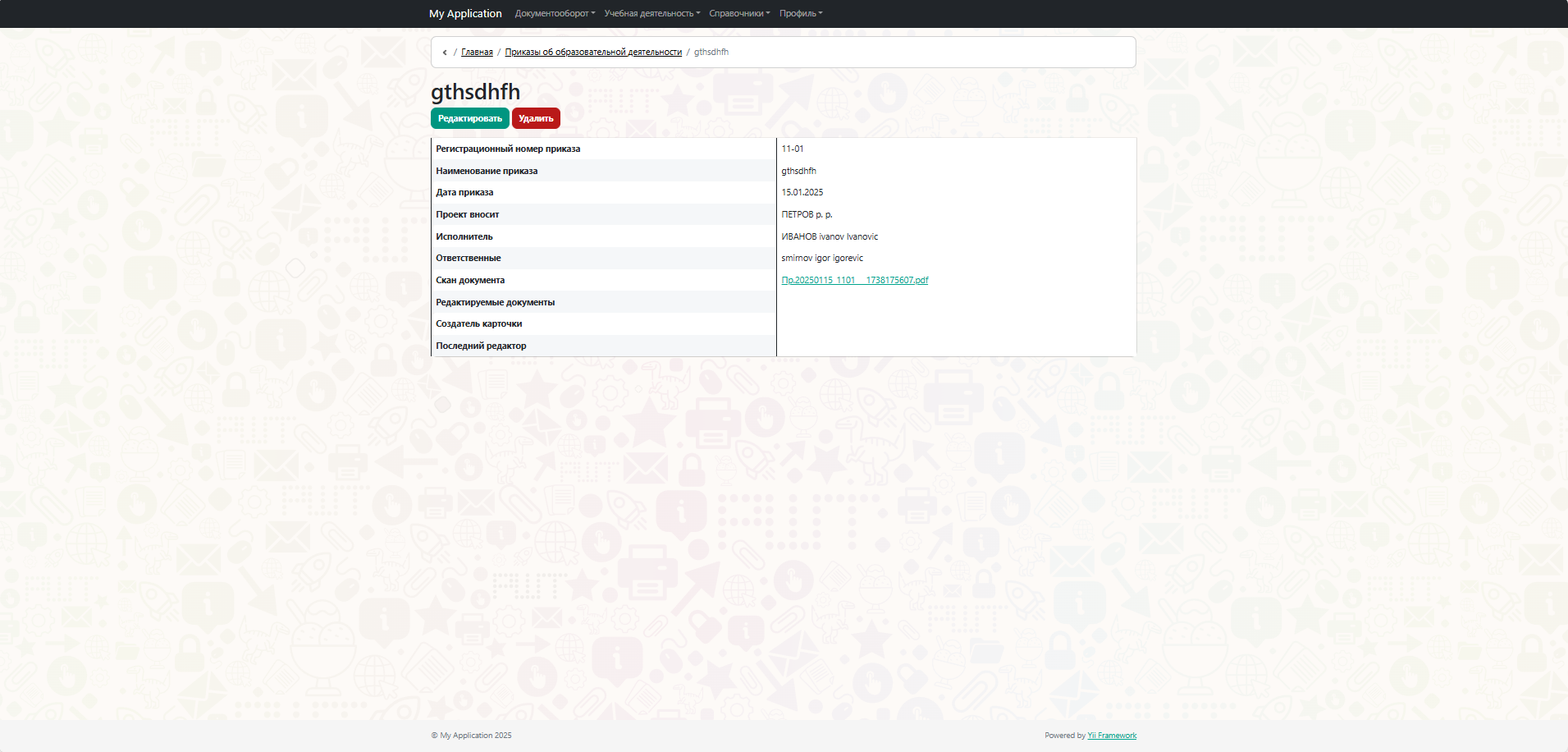


Рисунок 1.7 - Интерфейс view

* 1. **Требование к техническому и программному обеспечению.**

Требование к программному обеспечению клиента:

* ОС – Windows 10.
* Microsoft Edge 20.10240 и выше.
* Mozilla Firefox версии 41.0 и выше.
* Google Chrome версии 45.0 и выше.

Требования к техническому обеспечению клиента:

* IBM-совместимый компьютер с МП Intel Pentium III или AMD K6 с тактовой частотой 500 МГц и выше;
* не менее 2 ГБ ОЗУ;
* клавиатура;
* манипулятор «мышь».

Требование к программному обеспечению сервера:

* ОС – Cent OS 7 / Linux Debian 11.
* MariaDB версии 10.3 и выше.
* Версия интерпретатора PHP 7.4 и выше.

Требования к техническому обеспечению клиента:

* IBM-совместимый компьютер с МП Intel Pentium III или AMD K6 с тактовой частотой 500 МГц и выше;
* не менее 4 ГБ ОЗУ;
* НЖМД со свободным со свободным объемом не менее 1 ГБ

1. **ТЕХНИЧЕСКИЙ ПРОЕКТ** 
   1. **Общие сведения о работе системы.**

Клиентская часть программного продукта разработана в текстовом редакторе PhpStorm (версия 2024.1) на языках HTML5 и JavaScript (версия ECMAScript 6+) с использованием веб-браузеров Mozilla Firefox (версии 100.0.2) и Google Chrome (версия 101.0.4951.67).

Серверная часть программного продукта разработана с применением аналогичных инструментов на языке PHP (версия 7.4) и фреймворке Yii2 Advanced для реализации веб-сервера, а также с использованием различных библиотек для реализации функционала (DynamicForm, GridVIew и т.д.). Для хранения данных используется база данных MariaDB (версия 10.3).

Работоспособность серверной части программы была проверена под управлением операционных систем Windows 10 v21H2 и Linux Debian 11 Bullseye

* 1. **Описание архитектуры системы**

В соответствии с проблематикой предметной области были выделены основные уровни распределения информации, которые отображены на диаграмме развертывания, представленной в Приложении 7.

Участник взаимодействует с сервером посредством веб-браузера. Изначально, веб-браузер отправляет запрос к веб-сервису сервера по протоколу «HTTPS» для загрузки файлов, необходимых для работы веб-клиента: HTML-страницы и скриптовые файлы формата «.js».

После того, как веб-клиент был загружен с сервера, пользователь может начинать работать в системе. Для этого веб-клиент участника отправляет запросы веб-сервису сервера и проходит процесс авторизации и аутентификации. Также через веб-сервис осуществляется загрузка и скачивание файлов на сервере.

На сервере, веб-сервис взаимодействует с компонентами репозиториев и сервисов данных в тех случаях, когда необходимо загрузить или сохранить данные, связанных с серверным состоянием, например в случае, когда пользователю необходимо прикрепить скан-копии приказов, или когда пользователь создал, удалил или отредактировал документацию.

* 1. **Инсталяция и выполнение программного продукта**

Для работоспособности программного продукта необходимо:

1. Необходимо склонировать репозиторий в целевую папку командой:

git clone https://github.com/4eshir/docs2.git

1. После клонирования системы из git, необходимо открыть консоль, перейти в корень проекта и выполнить команду:

php init

1. Затем выбрать режим Development и подтвердить выбор. После этого yii соберет стартовые файлы конфигурации приложений.
2. Создать файлы:

\_\_autocomplete.php

bootstrap.php

params.php

test.php

1. В файле common/config/main.php создать секцию params и установить параметр mainCompanyId в 1 (если идентификатор уже существует - то указать его). Данный ID является идентификатором РШТ как компании в БД.
2. Также в созданной секции необходимо установить параметр yandexApiKey в значение y0\_AgAEA7qkEK7HAAn5LwAAAADkMhh1CPjqd4DtS52DG7Vyd3i0JNf-NxY или другой ключ доступа к Яндекс.Диску через API.
3. Для удобного автодополнения в IDE можно добавить в конфигурационные файлы информацию о компонентах приложения.
4. Запустить консоль и выполнить команду инициализации:

php yii fill/init

1. Создать базу данных с именем docs2\_db
2. Установить необходимые параметры в common/config/main-local.php.
3. Применить все миграции
   1. **Описание программы**

В таблице 2.1 представлено описание класса DocumentOrderWork, организующего работу пользователя с приказами.

Таблица 2.1 – описание класса DocumentOrderWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| scanFile | поле | Представляет собой сканированную копию документа. |
| docFiles | поле | Представляет собой отредактированную копию документа. |
| appFIles | поле | Представляет собой файлы-приложения к документации. |
| getFileLinks() | функция | Обеспечивает доступ к загруженным файлам приказов. |
| attributeLabels() | функция | Задаёт пользовательские названия элементам таблицы document\_order. |
| rules() | функция | Определяет правила работы с классом. |

В таблице 2.2 представлено описание класса DocumentInWork, организующего работу пользователя с входящей документацией.

Таблица 2.2 – описание класса DocumentInWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| scanFile | поле | Представляет собой сканированную копию документа. |
| docFiles | поле | Представляет собой отредактированную копию документа. |
| appFIles | поле | Представляет собой файлы-приложения к документации. |
| needAnswer | поле | Указывает на необходимость указать ответ к документации. |
| dateAnswer | поле | Указывает на крайний срок ответа |
| attributeLabels() | функция | Задаёт пользовательские названия элементам таблицы document\_in. |
| rules() | функция | Определяет правила работы с классом. |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| getFullName() | функция | Отображает полное название документа. |
| getFullNumber() | функция | Отображает полный номер документа с постфиксом. |
| getFileLinks() | функция | Обеспечивает доступ к загруженным файлам документации. |
| getNeedAnswerString() | функция | Отображает «документ-ответ». |
| generateDocumentNumber() | функция | Генерирует номер документа в системе. |

В таблице 2.3 представлено описание класса DocumentOutWork, организующего работу пользователя с исходящей документацией.

Таблица 2.3 – описание класса DocumentOutWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| scanFile | поле | Представляет собой сканированную копию документа. |
| docFile | поле | Представляет собой отредактированную копию документа. |
| appFIle | поле | Представляет собой файлы-приложения к документации. |
| isAnswer | поле | Указывает на то, является ли данный документ ответом на другой документ. |
| dateAnswer | поле | Указывает на крайний срок ответа. |
| attributeLabels() | функция | Задаёт пользовательские названия элементам таблицы document\_in. |
| rules() | функция | Определяет правила работы с классом. |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| getFullName() | функция | Отображает полное название документа. |
| getFullNumber() | функция | Отображает полный номер документа с постфиксом. |
| getFileLinks() | функция | Обеспечивает доступ к загруженным файлам документации. |
| getIsAnswer() | функция | Отображает название документа, на который был дан ответ. |
| generateDocumentNumber() | функция | Генерирует номер документа в системе. |

В таблице 2.4 представлено описание класса TrainingGroupParticipantWork, организующего работу пользователя с обучающимися в учебных группах.

Таблица 2.4 – описание класса TrainingGroupParticipantWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| generateDocumentNumber() | функция | Генерирует уникальный номер группы в системе. |
| getFileLinks() | функция | Обеспечивает доступ к загруженным файлам группы. |
| getActivity() | функция | Проверяет взаимодействие группы с приказами. |
| getActualGroup() | функция | Проверяет состоит ли участник в группе |

В таблице 2.5 представлено описание класса OrderTrainingGroupParticipantWork.

Таблица 2.5 – описание класса OrderTrainingGroupParticipantWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |

В таблице 2.6 представлено описание класса TrainingGroupWork, обеспечивающего работу с учебными группами.

Таблица 2.6 – описание класса TrainingGroupWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| generateNumber() | функция | Генерирует уникальный номер группы. |
| getFileLinks() | функция | Обеспечивает доступ к загруженным файлам группы. |

В таблице 2.7 представлено описание класса EventWork, реализующего внутренние мероприятия «Регионального школьного технопарка».

Таблица 2.7 – описание класса EventWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| attributeLabels() | функция | Задаёт пользовательские названия элементам таблицы document\_in. |
| rules() | функция | Определяет правила работы с классом. |
| getFileLinks() | функция | Обеспечивает доступ к загруженным файлам мероприятия. |
| getOrderNameRaw() | функция | Получает связанный с мероприятием приказ. |
| checkFilesExist() | функция | Проверяет на наличие файлы, связанные с мероприятием. |

В таблице 2.8 представлено описание класса ForeignEventWork, реализующего внешние мероприятия, в которых участвуют обучающиеся «Регионального школьного технопарка».

Таблица 2.8 – описание класса ForeignEventWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| getTeachers() | функция | Показывает задействованных педагогов в мероприятии. |

В таблице 2.9 представлено описание класса PeopleWork, реализующего физическое лицо (человека), который взаимодействует с «РШТ».

Таблица 2.9 – описание класса PeopleWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| getFullFio() | функция | Показывает полное имя человека. |
| getPositionName() | функция | Показывает должность человека в организации(если она есть) |

В таблице 2.10 представлено описание класса PeopleStampWork, реализующего связь «человек-ответственность»

Таблица 2.10 – описание класса PeopleStampWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |

В таблице 2.11 представлено описание класса ExpireWork, реализующего влияние приказа об основной деятельности на положение.

Таблица 2.11 – описание класса ExpireWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| getStatus() | функция | Указывает текущий статус. |
| getNumber() | функция | Указывает номер связанного приказа |

В таблице 2.12 представлено описание класса OrderMainWork, реализующего приказы об основной деятельности.

Таблица 2.11 – описание класса OrderMainWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| attributeLabels() | функция | Задаёт пользовательские названия элементам таблицы document\_in. |
| rules() | функция | Определяет правила работы с классом. |
| generateDocumentNumber() | функция | Генерирует номер приказа в системе. |

В таблице 2.13 представлено описание класса OrderEventWork, реализующего приказы о мероприятиях.

Таблица 2.13 – описание класса OrderEventWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| generateDocumentNumber() | функция | Генерирует номер приказа в системе. |

В таблице 2.14 представлено описание класса OrderTrainingWork, реализующего приказы об образовательной деятельности.

Таблица 2.14 – описание класса OrderTrainingWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| getNomenclature() | функция | Определяет статус номенклатуру приказа. |
| getStatus() | функция | Определяет статус приказа. |
| generateDocumentNumber() | функция | Генерирует номер приказа в системе. |

В таблице 2.15 представлено описание класса RegulationWork, реализующего приказы об образовательной деятельности.

Таблица 2.15 – описание класса RegulationWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| expires | поле | документ, отменяющий текущее положение |
| scanFile | поле | Представляет собой сканированную копию приложения. |
| \_construct() | функция | Создает новый экземпляр класса с заданными параметрами. |
| rules() | функция | Определяет правила работы с классом. |
| getFileLinks() | функция | Обеспечивает доступ к загруженным файлам положения. |

В таблице 2.16 представлено описание класса ActParticipantWork, реализующего акт участия в мероприятии.

Таблица 2.16 – описание класса ActParticipantWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| getTeam() | функция | Отображает команду участников. |
| getParticipants() | функция | Отображает участников акта об участии. |
| getFileLinks() | функция | Обеспечивает доступ к загруженным файлам акта об участии. |

В таблице 2.17 представлено описание класса SquadParticipantWork, реализующего участника в мероприятии.

Таблица 2.17 – описание класса SquadParticipantWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |

В таблице 2.18 представлено описание класса TeamNameWork, реализующего акт участия в мероприятии.

Таблица 2.18 – описание класса TeamNameWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |

В таблице 2.19 представлено описание класса ForeignEventParticipantsWork, реализующего участников образовательного процесса.

Таблица 2.19 – описание класса ForeignEventParticipantsWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| attributeLabels() | функция | Задаёт пользовательские названия элементам таблицы document\_in. |
| rules() | функция | Определяет правила работы с классом. |
| getFullFio() | функция | Показывает полное имя участников. |

В таблице 2.20 представлено описание класса ActParticipantBranchWork, реализующего связь «акт участия - отдел».

Таблица 2.20 – описание класса ActParticipantBranchWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |

В таблице 2.21 представлено описание класса FilesWork, реализующего работу с файлами системы.

Таблица 2.21 – описание класса FilesWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |

В таблице 2.22 представлено описание класса OrderPeopleWork, реализующего работу с файлами системы.

Таблица 2.22 – описание класса OrderPeopleWork

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| getFullFio() | функция | Показывает полное имя ответственного. |

В таблице 2.23 представлено описание класса PeopleStampService, реализующего бизнес-логику класса PeopleStampWork.

Таблица 2.23 – описание класса PeopleStampService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| createStampFromPeople() | функция | Реализует связи вида «человек-ответственность» |

В таблице 2.24 представлено описание класса ActParticipantService, реализующего бизнес-логику класса ActParticipantWork.

Таблица 2.24 – описание класса ActParticipantService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getFileInstance() | функция | Получает файл из POST-запроса. |
| saveFilesFromModel() | функция | Сохраняет файла в каталогах системы и в БД. |
| addActParticipant() | функция | Создаёт акт об участии |
| createActTable() | функция | Создаёт таблицу, содержащую информацию об актах. |
| createActFileTable() | функция | Создаёт таблицу, содержащую информацию об файлах актов. |

В таблице 2.25 представлено описание класса SquadParticipantService, реализующего бизнес-логику класса SquadParticipantWork.

Таблица 2.25 – описание класса SquadParticipantService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| addSquadParticipantEvent() | функция | Получает файл из POST-запроса. |
| updateSquadParticipantEvent() | функция | Сохраняет файл в каталогах системы и в БД. |

В таблице 2.26 представлено описание класса ForeignEventParticipantsService, реализующего бизнес-логику класса ForeignEventParticipantsWork.

Таблица 2.26 – описание класса ForeignEventParticipantsService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| checkCorrectAll() | функция | Проверяет корректности данных об ученике в группе. |
| checkCorrectOne() | функция |

В таблице 2.27 представлено описание класса PeopleService, реализующего бизнес-логику класса PeopleWork.

Таблица 2.27 – описание класса PeopleService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| createPositionsCompaniesArray() | функция | Создает связь типа «человек-должность-компания» |
| isAvailableDelete() | функция | Возвращает список ошибок, если список пуст - проблем нет |

В таблице 2.28 представлено описание класса DocumentInService, реализующего бизнес-логику класса DocumentInWork.

Таблица 2.28 – описание класса DocumentInService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getFileInstance() | функция | Получает файл из POST-запроса. |
| saveFilesFromModel() | функция | Сохраняет файла в каталогах системы и в БД. |
| getUploadedFilesTables() | функция | Получает все файлы документа в виде таблицы. |
| getPeopleStamps() | функция | Получает связь вида «человек-ответственность» |

В таблице 2.29 представлено описание класса DocumentOutService, реализующего бизнес-логику класса DocumentOutWork.

Таблица 2.29 – описание класса DocumentOutService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getFileInstance() | функция | Получает файл из POST-запроса. |
| saveFilesFromModel() | функция | Сохраняет файла в каталогах системы и в БД. |
| getUploadedFilesTables() | функция | Получает все файлы документа в виде таблицы. |
| getPeopleStamps() | функция | Получает связь вида «человек-ответственность» |

В таблице 2.30 представлено описание класса TrainingGroupService, реализующего бизнес-логику класса TrainingGroupWork.

Таблица 2.30 – описание класса TrainingGroupService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getFileInstance() | функция | Получает файл из POST-запроса. |
| saveFilesFromModel() | функция | Сохраняет файла в каталогах системы и в БД. |
| getUploadedFilesTables() | функция | Получает все файлы документа в виде таблицы. |
| attachTeachers() | функция | Прикрепляет к группе учителя. |
| attachThemes() | функция | Прикрепляет к группе учебную программу. |
| attachParticipants() | функция | Прикрепляет к группе учеников. |
| attachLessons() | функция | Прикрепляет к группе расписание занятий. |
| prepareFormScheduleData() | функция | Получает расписание в виде таблице. |

В таблице 2.31 представлено описание класса ForeignEventService, реализующего бизнес-логику класса ForeignEventWork.

Таблица 2.31 – описание класса ForeignEventService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getFileInstance() | функция | Получает файл из POST-запроса. |
| saveActFilesFromModel() | функция | Сохраняет файла в каталогах системы и в БД. |
| AttachAchievement() | функция | Прикрепляет достижения к мероприятию. |

В таблице 2.32 представлено описание класса DocumentOrderService, реализующего бизнес-логику класса DocumentOrderWork.

Таблица 2.32 – описание класса ForeignEventService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getFileInstance() | функция | Получает файл из POST-запроса. |
| saveFilesFromModel() | функция | Сохраняет файла в каталогах системы и в БД. |
| getUploadedFilesTables() | функция | Получает все файлы документа в виде таблицы. |
| deleteOrderMain() | функция | Удаляет из БД приказ об основной деятельности. |
| deleteOrderEvent() | функция | Удаляет из БД приказ о мероприятии. |
| deleteOrderTraining() | функция | Удаляет из БД приказ об образовательной деятельности. |

В таблице 2.33 представлено описание класса OrderMainService, реализующего бизнес-логику класса OrderMainWork.

Таблица 2.33 – описание класса OrderMainService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getUploadedFilesTables() | функция | Получает все файлы документа в виде таблицы. |
| createOrderNumber() | функция | Создает номер приказа. |

В таблице 2.34 представлено описание класса OrderTrainingService, реализующего бизнес-логику класса OrderTrainingWork.

Таблица 2.34 – описание класса OrderTrainingService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| isPossibleToInsertTrainingGroupParticipant() | функция | Проверяет возможность перевода/зачисления/отчисления |
| isPossibleToDeleteOrderTrainingGroupParticipant() | функция | Проверяет возможность отмены перевода/зачисления/отчисления |
| createOrderTrainingGroupParticipantEvent() | функция | Зачисляет/отчисляет/переводит ученика. |
| updateOrderTrainingGroupParticipantEvent() | функция | Изменяет положение ученика в рамках учебного процесса. |

В таблице 2.35 представлено описание класса RegulationService, реализующего бизнес-логику класса RegulationWork.

Таблица 2.35 – описание класса RegulationService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getFileInstance() | функция | Получает файл из POST-запроса. |
| saveFilesFromModel() | функция | Сохраняет файла в каталогах системы и в БД. |
| getUploadedFilesTables() | функция | Получает все файлы документа в виде таблицы. |

В таблице 2.36 представлено описание класса TeamService, реализующего бизнес-логику класса TeamNameWork.

Таблица 2.36 – описание класса TeamService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| teamNameCreateEvent() | функция | Создает команду-участника в мероприятии. |
| getNamesByForeignEventId() | функция | Получает названия команд, участвующих в данном мероприятии. |

В таблице 2.37 представлено описание класса ActParticipantBranchService, реализующего бизнес-логику класса ActParticipantBranchWork.

Таблица 2.37 – описание класса ActParticipantBranchService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| addActParticipantBranchEvent() | функция | Создает связь вида «акт участия-отдел» |

В таблице 2.38 представлено описание класса EventService, реализующего бизнес-логику класса EventWork.

Таблица 2.38 – описание класса EventService

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getFileInstance() | функция | Получает файл из POST-запроса. |
| saveFilesFromModel() | функция | Сохраняет файла в каталогах системы и в БД. |
| getUploadedFilesTables() | функция | Получает все файлы документа в виде таблицы. |

В таблице 2.39 представлено описание класса ActParticipantBranchRepository, реализующего взаимодействие с базой данных.

Таблица 2.39 – описание класса ActParticipantBranchRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| prepareDeleteByAct() | функция | Подготавливает удаление объекта класса. |
| prepareCreate() | функция | Подготавливает создание связи «акт-отдел». |

В таблице 2.40 представлено описание класса ActParticipantRepository, реализующего взаимодействие с базой данных.

Таблица 2.40 – описание класса ActParticipantRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getByForeignEventId() | функция | Получает список актов участия по мероприятию. |
| prepareCreate() | функция | Подготавливает создание акта об участии. |

В таблице 2.41 представлено описание класса SquadParticipantRepository, реализующего взаимодействие с базой данных.

Таблица 2.41 – описание класса SquadParticipantRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| prepareCreate() | функция | Подготавливает создание участника в команде мероприятия. |
| prepareDelete() | функция | Подготавливает удаление участника в команде мероприятия. |
| getAllFromEvent() | функция | Получает всех участвующих в мероприятии. |
| getCountByActAndParticipantId() | функция | Получает количество участников, участвующих в мероприятии. |

В таблице 2.42 представлено описание класса ForeignEventParticipantsRepository, реализующего взаимодействие с базой данных.

Таблица 2.42 – описание класса ForeignEventParticipantsRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getSortedList() | функция | Получает список участников. |
| prepareUpdate() | функция | Подготавливает изменение участника. |

В таблице 2.43 представлено описание класса PeopleRepository, реализующего взаимодействие с базой данных.

Таблица 2.43 – описание класса PeopleRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| prepareCreate() | функция | Подготавливает создание человека в системе |
| prepareDelete() | функция | Подготавливает удаление участника в команде мероприятия. |
| getOrderedList() | функция | Получает список людей |

В таблице 2.44 представлено описание класса PeopleStampRepository, реализующего взаимодействие с базой данных.

Таблица 2.44 – описание класса PeopleStampRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getStamps() | функция | Получает связь вида «человек-ответственность». |
| getSimilar() | функция | Получает все связи по человеку. |

В таблице 2.45 представлено описание класса InOutDocumentsRepository, реализующего взаимодействие с базой данных.

Таблица 2.45 – описание класса InOutDocumentsRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| prepareCreate() | функция | Подготавливает создание связи вида «входящий-исходящий» документ. |
| prepareUpdate() | функция | Подготавливает изменение связи вида «входящий-исходящий» документ. |
| prepareDelete() | функция | Подготавливает удаление связи вида «входящий-исходящий» документ. |

В таблице 2.46 представлено описание класса TrainingGroupRepository, реализующего взаимодействие с базой данных.

Таблица 2.46 – описание класса TrainingGroupRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getParticipants() | функция | Получает участников группы. |
| getLessons() | функция | Получает занятия группы. |
| getGroupsForCertificates() | функция | Получает сертификаты участников группы. |

В таблице 2.47 представлено описание класса TrainingGroupParticipantRepository, реализующего взаимодействие с базой данных.

Таблица 2.47 – описание класса TrainingGroupParticipantRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| prepareCreate() | функция | Подготавливает создание связи вида ученика в группе. |
| prepareUpdate() | функция | Подготавливает изменение связи вида ученика в группе. |
| prepareDelete() | функция | Подготавливает удаление связи вида ученика в группе. |
| getParticipantsToEnrollCreate() | функция | Получает учеников, которых можно зачислить в группу. |
| getParticipantsToDeductCreate() | функция | Получает учеников, которых можно отчислить из группы. |
| getParticipantsToTransferCreate() | функция | Получает учеников, которых можно перевести между группами. |
| getParticipantToEnrollUpdate() | функция | Получает учеников, которых можно зачислить в группу при изменении приказа. |
| getParticipantToDeductUpdate() | функция | Получает учеников, которых можно отчислить из группы при изменении приказа . |
| getParticipantToTransferUpdate() | функция | Получает учеников, которых можно перевести между группами при изменении приказа. |

В таблице 2.48 представлено описание класса OrderTrainingGroupParticipantRepository, реализующего взаимодействие с базой данных.

Таблица 2.48 – описание класса OrderTrainingGroupParticipantRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| prepareCreate() | функция | Подготавливает создание человека в системе |
| prepareDelete() | функция | Подготавливает удаление участника в команде мероприятия. |

В таблице 2.49 представлено описание класса FilesRepository, реализующего взаимодействие с базой данных.

Таблица 2.49 – описание класса FilesRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| prepareCreate() | функция | Подготавливает создание объекта файла в системе. |
| prepareUpdate() | функция | Подготавливает изменение объекта файла в системе. |
| prepareDelete() | функция | Подготавливает удаление объекта файла в системе. |

В таблице 2.50 представлено описание класса OrderPeopleRepository, реализующего взаимодействие с базой данных.

Таблица 2.50 – описание класса OrderPeopleRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getResponsiblePeople() | функция | Получает ответственных за определённые задачи людей. |
| prepareCreate() | функция | Подготавливает создание связи «ответственный-приказ» |
| prepareDelete() | функция | Подготавливает удаление связи «ответственный-приказ» |

В таблице 2.51 представлено описание класса TeamRepository, реализующего взаимодействие с базой данных.

Таблица 2.51 – описание класса TeamRepository

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| getByNameAndForeignEventId() | функция | Получает уникальный обьект класса TeamNameWork по имени. |
| getNamesByForeignEventId() | функция | Получает список команд по мероприятию. |

**ПРОЧИЕ КЛАССЫ:**

* **Events**
* **FilenameGenerator**
* **Helper**
  1. **Разработанные меню и ннтерфейсы.**
  2. **Сообщения системы.**

# **ЗАКЛЮЧЕНИЕ**

В процессе выполнения курсового проекта были решены следующие задачи:

* изучена предметная область;
* разработаны различные диаграммы, иллюстрирующие работу системы;
* написан рабочий проект.

Дальнейшее развитие информационной системы:

* разработка программного продукта;
* написание рабочего проекта;
* разработка дизайна форм.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Левитин А. В. Алгоритмы. Введение в разработку и анализ — 2006.
2. Белов С.В., Лаптев В.В., Морозов А.В., Толасова В.В., Мамлеева А.Р. Требования к оформлению студенческих работ. / АГТУ – Астрахань, 2019.
3. Алгоритмы и структуры данных — Кормен – 2022.
4. “Грокаем алгоритмы. Иллюстрированное пособие для программистов и любопытствующих” - Адитья Бхаргава (2022) Издательство: Литрес.
5. “Чистый код. Создание, анализ и рефакторинг” Мартин Роберт С. Издательство: Питер (2022).
6. Дэвид Скляр "Изучаем PHP 7. Руководство по созданию веб-сайтов" (2017).
7. «Yii. Сборник рецептов» Автор: Макаров А. (2012).
8. «Web Application Development with Yii 2 and PHP» Авторы: Mark Safronov и Jeffrey Winesett.
9. «Чистая архитектура. Искусство разработки программного обеспечения» - Мартин Роберт С. Издательство: Питер (2022).
10. «Современный подход к программной архитектуре: сложные компромиссы» - Нил Форм, Марк Ричардс (2023).

**ПРИЛОЖЕНИЕ 1**

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**на разработку программного продукта**

**«Автоматизация сопровождения образовательного процесса в организации «Региональный школьный технопарк»»**

Направление: 09.03.01 Информатика и вычислительная техника

Исполнитель: обучающийся гр. ДИНРб-41 Кузургалиев Р.А.

1. **Назначение, цели и задачи разработки Цели разработки:**

* обеспечение сопровождения образовательного процесса и документооборота в организации «Региональный школьный технопарк».

**Назначение разработки:** система предназначена для бесперебойной работы системы автоматизации сопровождения образовательного процесса.

**Основные задачи,** решаемые разработчиком в процессе разработки системы:

* анализ предметной области;
* разработка программного продукта в соответствии с требованиями;
* документирование проекта в соответствии с установленными требованиями.

1. **Характер разработки:** прикладная квалификационная работа.
2. **Основания для разработки**

* Учебный план направления 09.03.01 «Информатика и вычислительная техника» 20\_\_ года набора.
* Распоряжение по кафедре АСОИУ № от « » 20 г.

1. **Плановые сроки выполнения** Начало « » 20 г. Окончание « » 20 г.
2. **Требования к проектируемой системе**
   1. **Требования к функциональным характеристикам**

Проектируемая система должна обеспечивать выполнение следующих основных функций:

* Создавать учебные группы и прикреплять к ней учеников.
* Автоматически привязывать приказы об образовательной деятельности к образовательным программам и группам.
* Добавлять информацию о мероприятиях, проводимых в «РШТ», так за её пределами.
* Добавлять и удалять файлы в системе.
* Проводить аналитику образовательного процесса.
* Изменять статусы учеников в системе.
  1. **Требования к эксплуатационным характеристикам:**
* Программа не должна аварийно завершаться при любых действиях пользователя.
* Время реакции программы на действия пользователя не должно превышать 15 секунд.
  1. **Требования к программному обеспечению:**

Операционная система сервера: Linux Debian 11 Bullseye или Windows 10.

Веб-браузер клиента: Firefox последней мажорной версии (на данный момент 100.x.x), Google Chrome последней мажорной версии (на данный момент 100.x.x.x).

Операционная система клиента: дистрибутивы Linux, в которых есть возможность установить браузер необходимой версии (см. выше), а также Windows 7 или более поздние.

* 1. **Требования к аппаратному обеспечению:**

Рекомендуемая конфигурация для сервера:

* Процессор: не менее 2.4 ГГц, 4 ядра.
* Оперативная память: не менее 4 Гб.
* Свободное место на жестком диске: не менее 1 Гб.
* Минимальная скорость входящего подключения: 1 Мбит/с.
* Минимальная скорость исходящего подключения: 1 Мбит/с.

Рекомендуемая конфигурация для клиента:

* Процессор Intel Pentium 3 или старше с поддержкой SSE3.
* Оперативная память: не менее 2 Гб.
* Минимальная скорость входящего подключения: 512 Кбит/с.
* Минимальная скорость исходящего подключения: 512 Кбит/с.

1. **Стадии и этапы разработки**
   1. **Эскизный проект (ЭП)**

* Анализ предметной области.
* Подготовка проектной документации.
  1. **Технический проект (ТП)**
* Разработка структур и форм представления данных.
* Разработка структуры программного комплекса.
* Подготовка пояснительной записки.
  1. **Рабочий проект (РП)**
* Программная реализация.
* Тестирование и отладка программы.
* Подготовка программной и эксплуатационной документации.
  1. **Эксплуатация (Э)**

Описание и анализ результатов проведенного исследования.

1. **Требования к документированию проекта**

К защите выпускной квалификационной работы должны быть представлены следующие документы:

* Пояснительная записка к ВКР.
* Презентация доклада.
* Программа, презентация и пояснительная записка к ВКР на оптическом носителе.

Требования к структуре документов определены соответствующими стандартами ЕСПД.

Требования к оформлению определены соответствующими методическими указаниями.

1. **Порядок контроля и приемки**

Контроль выполнения выпускной квалификационной работы проводится научным руководителем поэтапно в соответствии с утвержденным графиком выполнения ВКР.

На завершающем этапе руководитель осуществляет нормоконтроль представленной исполнителем документации и принимает решение о допуске (недопуске) ВКР к защите.

В процессе защиты ВКР исполнитель представляет документацию, делает краткое сообщение по теме разработки и демонстрирует ее программную реализацию.

При выставлении оценки учитывается:

* степень соответствия представленной разработки требованиям технического задания;
* качество программной реализации, документации и доклада по теме проекта;
* соблюдение исполнителем графика выполнения ВКР.

**ПРИЛОЖЕНИЕ 2**

**ДИАГРАММА КЛАССОВ**

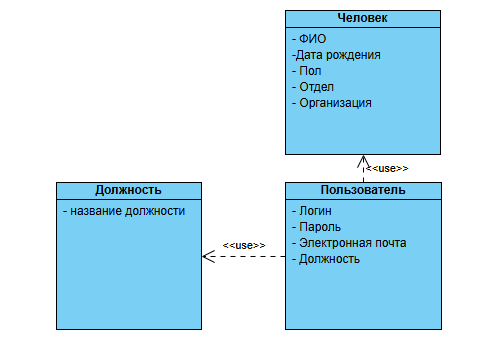


Рисунок П2.1 – Диаграмма классов Человек, Пользователь, Должность

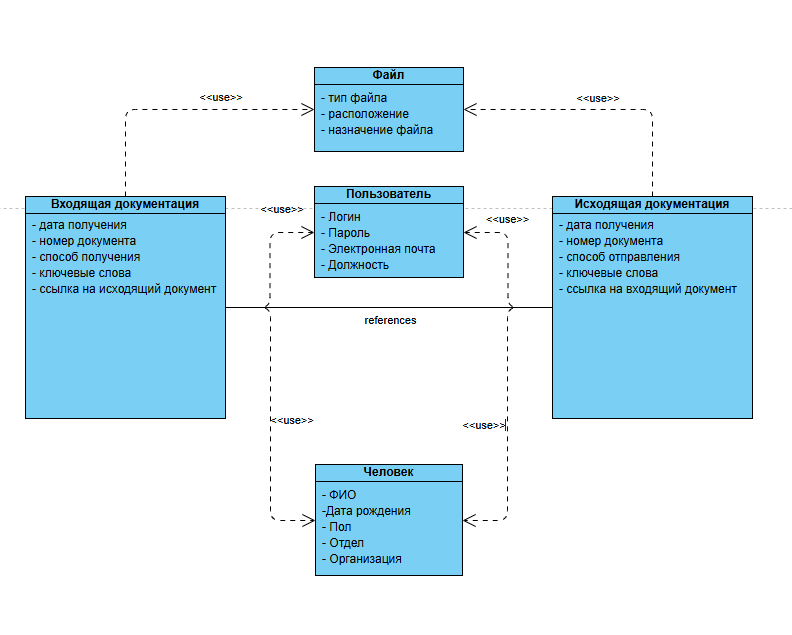


Рисунок П2.2 – Диаграмма классов Входящая и Исходящая документация

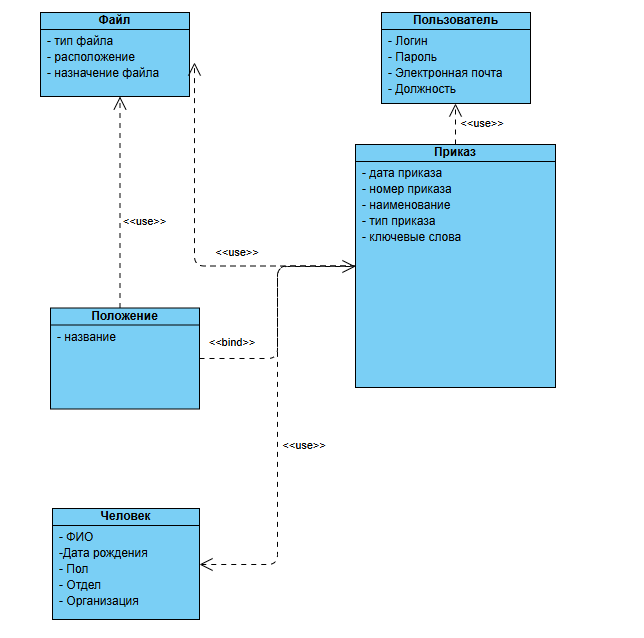


Рисунок П2.3 – Диаграмма классов Приказ, Положение

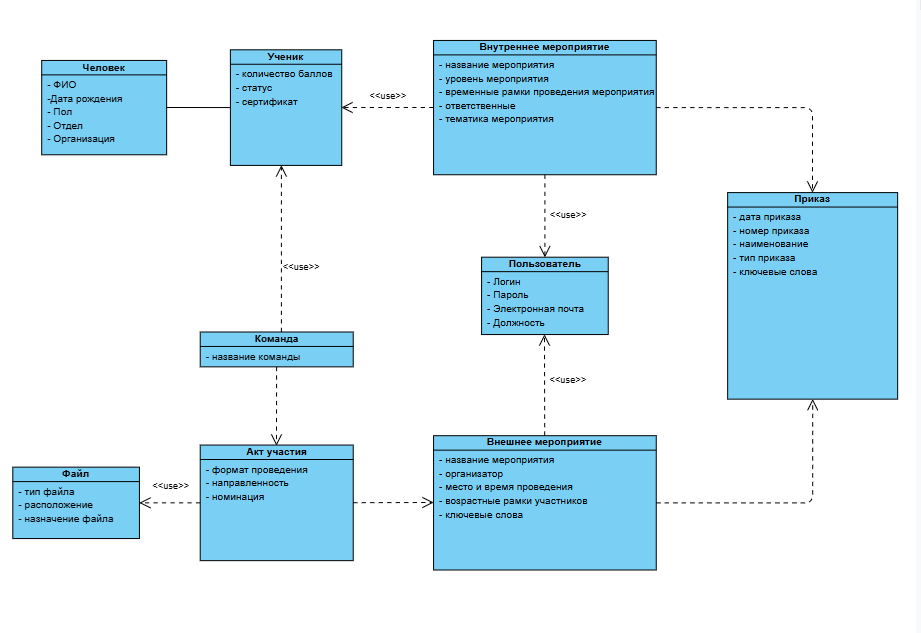


Рисунок П2.4 – Диаграмма классов Мероприятие, Акт участия, Команда

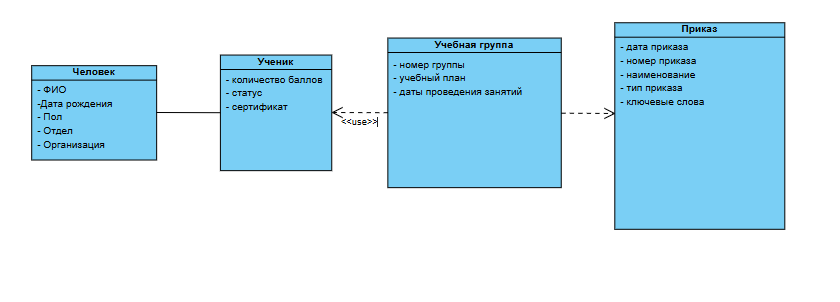


Рисунок П2.5 – Диаграмма классов Учебная группа, Ученик

**ПРИЛОЖЕНИЕ 3**

**ФУНКЦИОНАЛЬНЫЕ ДИАГРАММЫ IDEF0**

Рисунок П3.1 - Функциональная диаграмма «Обработка документации»

Рисунок П3.2 - Функциональная диаграмма «Приказы о мероприятии»

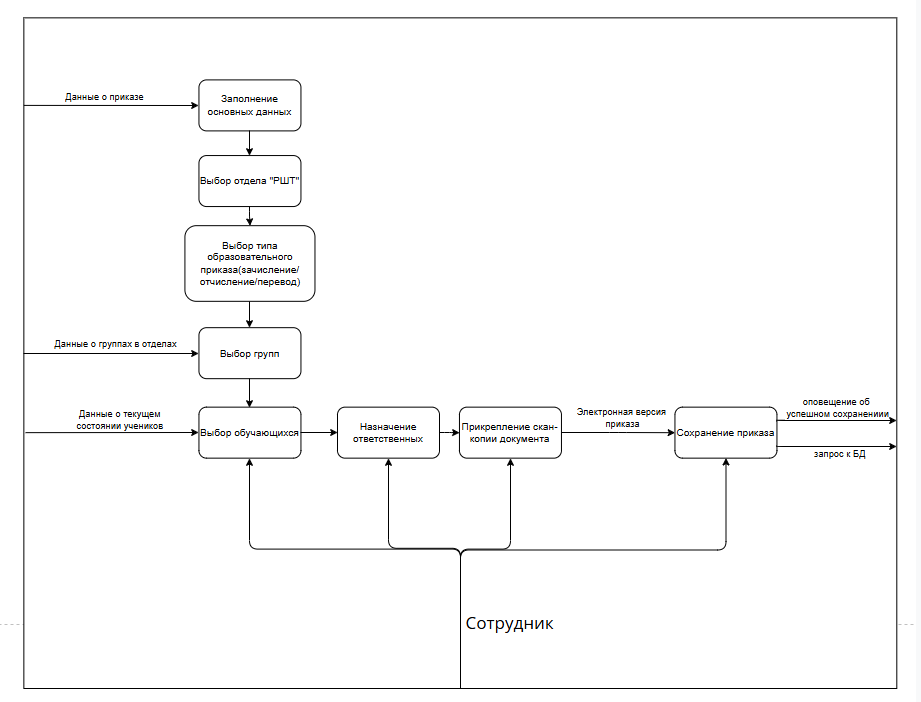


Рисунок П3.3 - Функциональная диаграмма «Приказы об образовательной деятельности»

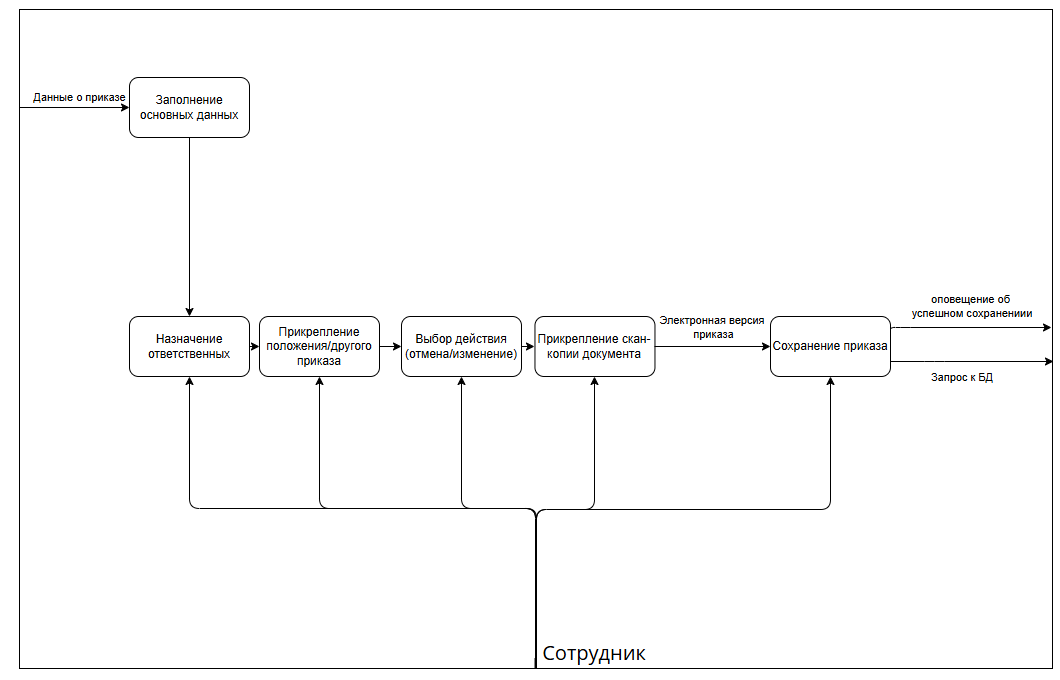


Рисунок П3.4 - Функциональная диаграмма «Приказы об основной деятельности»

**ПРИЛОЖЕНИЕ 4**

**ДИАГРАММА СОСТОЯНИЙ**

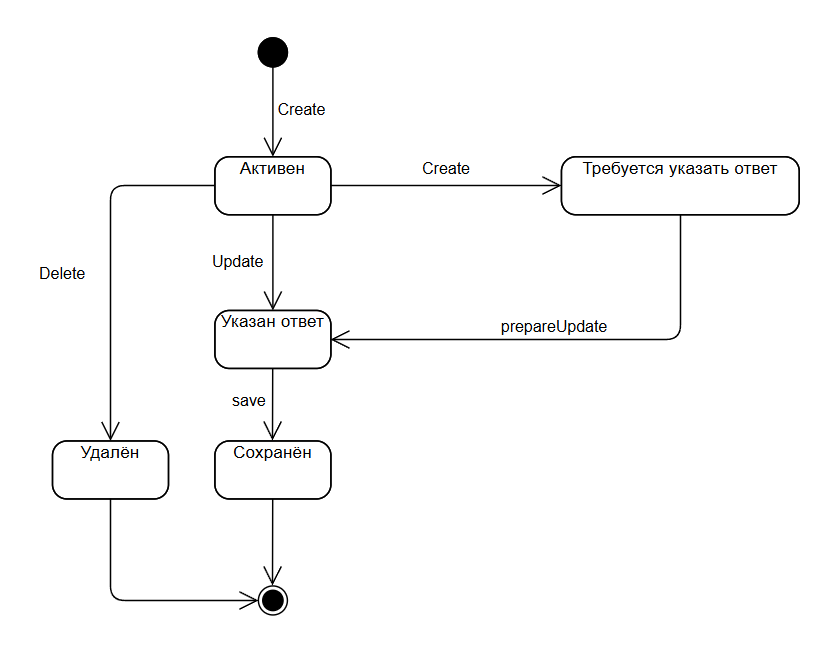


Рисунок П4.1 – Диаграмма состояний Входящая и Исходящая документация

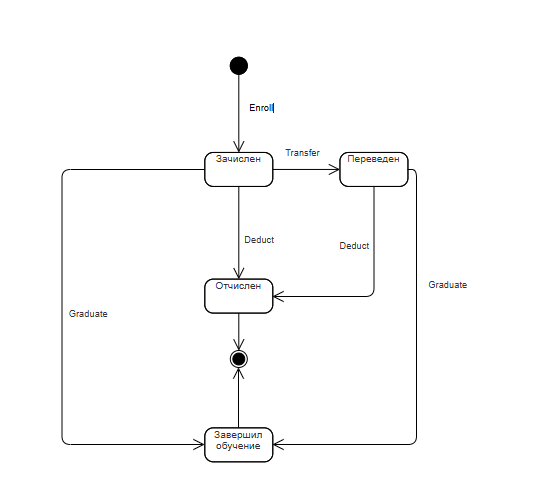
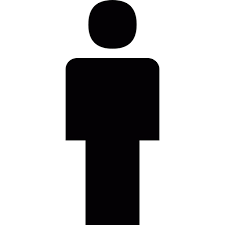


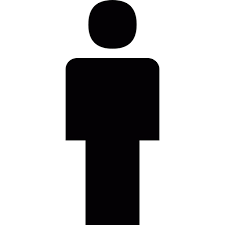
Рисунок П4.2 – Диаграмма состояний Ученик.

**ПРИЛОЖЕНИЕ 5**

**ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ**



Педагог



Администратор

Рисунок П5.1 - Диаграмма вариантов использования