|  |  |  |
| --- | --- | --- |
| лого для документов 2022 | ***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**  **ООО «ДКС РУС» по международному стандарту ISO 9001:2015** | |
| Институт информационных технологий и коммуникаций  Направление подготовки: 09.03.01 Информатика и вычислительная техника  Профиль: Автоматизированные системы обработки информации и управления  Кафедра «Автоматизированные системы обработки информации и управления» | | |
| **КУРСОВОЙ ПРОЕКТ**  **Автоматизация планирования образовательного процесса в организации «Региональный школьный технопарк»**  по дисциплине «Проектирование и архитектура автоматизированных систем» | | |
| Допущен к защите:  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г.  Руководитель:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Оценка, полученная на защите: «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»  Дата защиты: \_\_\_\_\_\_\_\_\_\_\_\_\_  Члены комиссии:  Морозов А.В. \_\_\_\_\_\_\_\_\_\_\_\_\_ | | Проект выполнил  обучающийся группы ДИНРБ-41 Кузургалиев Р.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Руководитель  к.т.н., доцент, Морозов А.В. |

**Астрахань – 2024**

**ФЕДЕРАЛЬНОЕ АГЕНТСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ ПО РЫБОЛОВСТВУ**

**АСТРАХАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

|  |  |
| --- | --- |
| **УТВЕРЖДАЮ** | Кафедра «Автоматизированные системы  обработки информации и управления» |
| Заведующий кафедрой  д.т.н., профессор  Т.В.Хоменко\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

**ЗАДАНИЕ**

**на выполнение курсового проекта**

Обучающийся ***Кузургалиев Радмир Алексеевич***

Группа ***ДИНРБ-41***

Дисциплина ***Проектирование и архитектура автоматизированных систем***

Тема ***Автоматизация планирования образовательного процесса в организации «Региональный школьный технопарк»***

Дата получения задания «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024г.  
Срок представления обучающимся КП на кафедру «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_2024г.

Руководитель ***к.т.н., доцент*** \_\_\_\_\_\_\_\_\_ ***Морозов А.В.*** «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_2024г.

*степень звание подпись ФИО*

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ***Кузургалиев Р.А.*** «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_2024г.

*подпись ФИО*

**Задачи**

|  |  |  |
| --- | --- | --- |
|  | 1 | Создание технического проекта: |
|  |  | - анализ предметной области; |
|  |  | - анализ аналогов; |
|  |  | - разработка диаграмм и схем. |
|  |
|  |
|  |
|  |

|  |  |
| --- | --- |
| **УТВЕРЖДАЮ** | К заданию на курсовой проектпо дисциплине  «Проектирование и архитектура автоматизированных систем» |
| Заведующий кафедрой  д.т.н., профессор  Т.В. Хоменко \_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

**КАЛЕНДАРНЫЙ ГРАФИК**

выполнения курсового проекта

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Разделы, темы и их содержание, графический материал | Дата сдачи | Объем, % |
| 1 | Утверждение темы курсового проекта |  | 1 |
| 2 | Изучение предметной области.  Описание технологии обработки информации в системе.  Обзор аналогичных систем.  Разработка диаграммы вариантов использования. Определение цели и назначения системы. |  | 10 |
| 3 | Инфологическое проектирование базы данных.  Приведение базы данных к третьей нормальной форме. |  | 35 |
| 4 | Даталогическое проектирование базы данных в MariaDB.  Обеспечение целостности данных. |  | 50 |
| 5 | Оформление пояснительной записки и презентации. Предоставление пояснительной записки на проверку. |  | 80 |
| 6 | Предзащита: презентация курсового проекта, демонстрация базы данных и программного продукта. |  | 90 |
| 7 | Защита курсового проекта (по графику) |  | 60-100 |

С графиком ознакомлен «\_\_»\_\_\_\_\_\_\_\_2024г.

**Кузургалиев Р.А.** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, обучающийся группы ДИНРБ-41

*(фамилия, инициалы, подпись*)

График курсового проектирования выполнен без отклонений / с незначительными отклонениями / со значительными отклонениями

*нужное подчеркнуть*

Руководитель курсового проекта\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ к.т.н., доцент **Морозов А.В.**

**Содержание**

[**ВВЕДЕНИЕ** 6](#_Toc187578099)

[**1** **ТЕХНИЧЕСКИЙ ПРОЕКТ** 7](#_Toc187578100)

[**1.1** **Предметная область** 7](#_Toc187578101)

[**1.1.1** **Управление доступом** 11](#_Toc187578102)

[**1.1.2** **Технология хранения файлов в системе** 12](#_Toc187578103)

[**1.2** **Анализ существующей системы** 12](#_Toc187578104)

[**1.3** **Аналоги.** 14](#_Toc187578105)

[**1.3.1** **ФГИС «Моя школа»** 14](#_Toc187578106)

[**1.3.2** **«Сферум»** 15](#_Toc187578107)

[**1.3.3** **«Google Класс»** 15](#_Toc187578108)

[**1.3.4** **Сравнительная таблица аналогов** 16](#_Toc187578109)

[**1.3.5** **Достоинства и недостатки аналогов** 17](#_Toc187578110)

[**1.4** **Цель создания автоматизированной (информационной) системы.** 18](#_Toc187578111)

[**1.5** **Назначение автоматизированной (информационной) системы.** 18](#_Toc187578112)

[**1.6** **Обоснование выбора инструментов.** 18](#_Toc187578113)

[**1.7** **Технология обработки информации** 20](#_Toc187578114)

[**1.7.1** **Диаграмма вариантов использования** 20](#_Toc187578115)

[**1.7.2** **Диаграмма классов** 20](#_Toc187578116)

[**1.7.3** **Алгоритм добавления входящей и исходящей документации** 23](#_Toc187578117)

[**1.7.4** **Алгоритм добавления резерва** 24](#_Toc187578118)

[**1.7.5** **Алгоритм добавления приказа об основной деятельности** 24](#_Toc187578119)

[**1.7.6** **Алгоритм присвоения номера документа в системе** 24](#_Toc187578120)

[**1.7.7** **Алгоритм добавления приказа о мероприятии** 25](#_Toc187578121)

[**1.7.8** **Алгоритм добавления акта об участии в мероприятии** 25](#_Toc187578122)

[**1.7.9** **Алгоритм добавления образовательного приказа.** 25](#_Toc187578123)

[**1.7.10** **Алгоритм добавления образовательной группы.** 26](#_Toc187578124)

[**1.7.11** **Алгоритм зачисления и отчисления ученика в учебную группу.** 26](#_Toc187578125)

[**1.7.12** **Алгоритм перевода ученика между группами.** 26](#_Toc187578126)

[**1.8** **Основные сценарии работы программного продукта** 26](#_Toc187578127)

[**1.8.1** **Диаграмма состояний** 27](#_Toc187578128)

[**1.8.2** **Диаграмма активности** 27](#_Toc187578129)

[**1.9** **Требование к техническому и программному обеспечению.** 27](#_Toc187578130)

[**1.10** **Безопасность ИС.** 28](#_Toc187578131)

[**2** **РАБОЧИЙ ПРОЕКТ** 29](#_Toc187578132)

[**2.1** **Общие сведения о работе системы.** 29](#_Toc187578133)

[**2.2** **Описание архитектуры системы** 29](#_Toc187578134)

[**2.3** **Инсталяция и выполнение программного продукта** 29](#_Toc187578135)

[**2.4** **Описание программы** 30](#_Toc187578136)

[**ЗАКЛЮЧЕНИЕ** 38](#_Toc187578137)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 39](#_Toc187578138)

[**ПРИЛОЖЕНИЕ 1** 40](#_Toc187578139)

[**ПРИЛОЖЕНИЕ 2** 44](#_Toc187578140)

[**ПРИЛОЖЕНИЕ 3** 45](#_Toc187578141)

[**ПРИЛОЖЕНИЕ 4** 46](#_Toc187578142)

[**ПРИЛОЖЕНИЕ 5** 54](#_Toc187578143)

[**ПРИЛОЖЕНИЕ 6** 55](#_Toc187578144)

**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 эффективными инструментами для автоматизированного планирования 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 эффективность 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 оптимизации образовательных программ.

Однако в данный момент организация эксплуатирует информационную систему, которая не способна эффективно планировать образовательный процесс.

Целью разработки информационной системы для организации "Региональный школьный технопарк" является повышение эффективности планирования образовательного процесса и документооборота в организации «Региональный школьный технопарк».

Назначение автоматизированной (информационной) системы «Регионального школьного технопарка» заключается в обеспечении бесперебойной и эффективной работы системы автоматизации планирования 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**

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 образовательными 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.

Таким образом, разработка информационной системы для организации "Региональный школьный технопарка необходима, так как она способствует повышению эффективности, улучшению обслуживания учащихся и созданию более конкурентоспособной и адаптивной образовательной структуры в условиях современной технической среды.

Можно выделить 2 основных процесса, 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 П2.1 в Приложении 2, 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 о мероприятиях.
3. Приказы об образовательной деятельности.

Диаграмма подпроцесса «Приказы о мероприятиях» представлена на рисунке П2.2 в приложении 2, 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.

В ходе работы «Регионального школьного технопарка» необходимо активно взаимодействовать с разнообразной документацией, включая входящие и исходящие документы. 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 эффективной 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 эффективность 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 эффективное функционирование учреждения, поддерживая порядок и дисциплину в административных аспектах.
* 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. 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. Основная цель — активное участие учеников в разнообразных событиях для повышения их навыков, расширения кругозора и развития социальных связей.

«Региональный школьный технопарк» активно взаимодействует с различными компаниями, организациями и учреждениями, что неизбежно приводит к возникновению документооборота, поэтому информационная система должна эффективно управлять документами, обеспечивать сохранность и конфиденциальность информации, а также облегчать процессы совместной работы и обмена данными между различными участниками. Такая система поможет упростить процессы взаимодействия и повысить эффективность работы «РШТ».

Образовательный процесс в «Региональном школьном технопарке» проводится в учебных группах. Учебные группы — это небольшие коллективы студентов, 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 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, отсюда возникает возможность перевода между группами.   
 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.

"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 конечном итоге, влияет на привлечение будущих студентов и партнеров.

* + 1. **Управление доступом**

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.

* + 1. **Технология хранения файлов в системе**

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. **Анализ существующей системы**

«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.

Становится очевидным, что такой объём информации невозможно обработать вручную, к тому же становится актуальной проблема планирования образовательного процесса. За эти годы были образованы

* 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.

«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, изменять статус обучающихся (переводить их между группами и отчислять).
  + Существующая система не обрабатывает информацию о мероприятиях, которые проводит «РШТ» и в которых участвуют ученики технопарка.

**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.

В рамках дипломной работы требуется разработать и внедрить информационную систему для организации «Региональный школьный технопарк» с учётом устранения вышеописанных недостатков.  По результатам анализа предметной области было определено, что необходимо разработать новую базу данных на основе существующей, которая будет содержать информацию о документации, расположении файлов в системе и образовательном процессе. Это включает в себя создание сущностей (таблиц) для хранения данных о документации, приказах, учебных группах и мероприятиях. Информационная система должна отвечать следующим 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 эффективного добавления и удаления файлов из системы.
* Аналитика образовательного процесса.
* Возможность изменения статусов учеников в системе.

На данный момент ни одна из систем-аналогов не соответствует в полной мере данным требованиям.

* 1. **Аналоги.** 
     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** | **Разрабатываемая система** |
| --- | --- | --- | --- | --- | --- |
| 1 | Ориентированность на организации дополнительного образования | - | - | + | + |
| 2 | Открытый исходный код | - | - | - | + |
| 3 | Возможность учёта учебных и внеучебных мероприятий | - | - | + | + |
| 4 | Учебные группы | + | + | + | + |
| 5 | Возможность учёта достижений | + | + | + | + |
| 6 | Аналитика образовательного процесса | + | + | + | + |

Таблица 1.1 - продолжение

| **№** | **Характеристики** | **«Моя школа»** | **«Сферум»** | **Google Класс** | **Разрабатываемая система** |
| --- | --- | --- | --- | --- | --- |
| 7 | Учёт 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 планирования 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 планирования 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 эффективной информационной системы, способной обеспечить планирование 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 эффективное 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.

* 1. **Назначение автоматизированной (информационной) системы.**

Назначение автоматизированной (информационной) системы «Регионального школьного технопарка» заключается в обеспечении бесперебойной и эффективной 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 эффективной, 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. **Обоснование выбора инструментов.**

Перед началом создания программного продукта 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.

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

Результатом анализа предметной области стала диаграмма вариантов использования, которая показана в Приложении 3 на рисунке П3.1. 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 образовательных программ.
* Просмотр участников деятельности.
* Получение уведомлений об ошибках в своих группах.
* Добавление новых учебных групп.
* Просмотр своих учебных групп.
* Редактирование своих учебных групп.

Вариант использования «Администратор» включает в себя следующие функции:

* Создать организацию.
* Создать сотрудника.
* Изменить роль сотрудника.
* Обработка документации.
* Обработка приказов.
  + 1. **Диаграмма классов**

В ходе анализа предметной области были разработаны диаграммы классов клиентской и серверной части программы. Они представлены в Приложении 4 на рисунках П4.1 – П4.11.

Класс **NomenclatureDictionary** содержит всю необходимую информацию о видах приказов и их кодовых обозначениях в организации.

Класс **EventFormDictionary** представляет информацию о виде проведения мероприятий.

Класс **EventLevelDictionary** представляет уровень проведения мероприятий.

Класс **EventTypeDictionary** содержит информацию о виде проводимых мероприятий.

Класс **EventWayDictionary** обозначает какой характер носит проводимое мероприятие.

Класс **FocusDictionary** олицетворяет направление, по которому проводится мероприятие.

Класс **DocumentTypeDictionary** содержит информацию о том, какие бывают разновидности документов.

Класс **BranchDictionary** описывает все подразделения «Регионального школьного технопарка», которые осуществляют образовательную деятельность.

Все вышеприведённые классы являются дочерними по отношению к классу **BaseDictionary**, который реализует основные методы, которые характерны для всех описанных сущностей.

Класс **DocumentIn** является центральной сущностью модуля работы с входящей документацией и включает в себя все соответствующие поля и методы по работе с документом, а класс **DocumentOut** отображает исходящую документацию в соответствующем модуле.

Класс **OrderMain** представляет всевозможные виды приказов в системе и содержит все основные методы их обработки, добавления и удаления. Подклассы **OrderEvent** и **OrderTraining** реализуют методы приказов о мероприятиях и учебной деятельности.

Класс **Regulation** реализует механизм положений, которые влияют на приказы, могут отменять и изменять их. Эту связь «приказ-положение» реализует класс **Expire.**

Класс **Event** предназначен для представления информации о мероприятии.

Класс **EventBranch** реализуют связь между мероприятием и отделом/отделами «РШТ», которое является ответственным за его проведение.

Класс **Position** отображает должность сотрудников «Регионального школьного технопарка».

Класс **People** предназначен для обозначения всех людей, которые имели взаимодействие с «Региональным школьным технопарком». К ним относятся сотрудники, ученики, представители власти и компаний-партнёров.

Класс **User** также предназначен для обозначения людей, но он реализует отображение пользователей системы, которые обладают определёнными правами и ролями.

Служебный класс **InOutDocument** реализует связь между входящей и исходящей документацией. Как было установлено после анализа программной области существуют случаю, когда на документ нужно ответить, именно эту связь реализует этот класс.

Служебный класс **PeoplePositionCompanyBranch** «прикрепляет» сотрудника к «РШТ» к должности и отделу, в котором он будет вести деятельность.

У каждого приказа есть хотя бы один человек, который ответственен за его выполнение. Класс **OrderPeople** реализует связь «ответственное лицо-приказ».

Класс **Files** отображает всю информацию о файлах, которые находятся в системе: их расположение и вид.

Классы **TeamName** содержит информацию о командах учеников «РШТ», которые участвуют в мероприятиях.

Задача класса **ActParticipant** — отображение связи «мероприятие-команда-ученик-номинация», а класса **SquadParticipant** — обеспечение связи между командой и участниками в рамках мероприятий. Сущность **ActParticipantBranch** отвечает за связь между актами участия в мероприятиях и ответственным отделом «РШТ».

Класс **ForeignEvent** отображает мероприятия, которые проводят организации, отличные от «Регионального школьного технопарка»

Классы **TrainingGroup** и **TrainingGroupParticipant** представляют учебные группы и их состав в рамках образовательного процесса. **OrderTrainingGroupParticipant** обеспечивает нормативно-правовую регуляцию учеников в группах.

Класс типа **helper** (или вспомогательный класс) — это класс, который предоставляет вспомогательные методы или функции для упрощения работы с основными классами или функциями.

Класс **FileHelper** необходим для обработки действий с классом Files, а именно распределение файла-скана документа в системе.

Класс **SortHelper** реализует различные методы сортировки, необходимые для оптимального взаимодействия с пользователем.

Класс **OrderNumberHelper** позволяет системе избежать коллизии номеров документов,

Классы **DocumentInFileNameGenerator. DocumentOutFileNameGenerator, OrderMainFileNameGenerator** и **FileNameGeneratorInterface** реализуют механизмы присвоения файлам в системе собственных имён, что необходимо для решения проблемы коллизии в системе.

Класс **DateFormater** реализует различные преобразования даты и времени.

Особым типом классов являются события (**event**), основная задача которых - обеспечение целостности системы, а также поддержание различных связей между классами. События **PeopleEventCreate, PeopleCompanyPositionBranchEventCreate, PeopleCompanyPositionBranchEventCreate** реализуют механизмы, необходимые для работы с классами **People** и **PeoplePositionCompanyBranch**.

Механизмы обработки связи между входящей и исходящей документации класса **InOutDocument** реализуют с помощью классов **InOutDocumentCreateEvent, InOutDocumentUpdateEvent, InOutDocumentLinkedEvent, InOutDocumentDeleteEvent**.

Класс **ExpireCreateEvent** реализует механизмы обеспечения целостности связи «приказ-положение».

Репозиторий (**Repository**) – это специализированный класс, отвечающий за доступ к данных. С ними взаимодействуют основные классы, а также их методы.

В системе реализованы следующие репозитории:

* **DocumentInRepository, DocumentOutRepository.**
* **RegulationRepository, OrderMainRepository, OrderTrainingRepository, OrderEventRepository.**
* **TrainingGroupRepository, TrainingGroupParticipantRepository, OrderTrainingGroupParticipantRepository.**
* **ExpireRepository, FilesRepository.**
* **PeopleRepository, OrderPeopleRepository, InOutDocumentRepository.**
* **TeamRepository, ActParticipantRepository.**
* **ForeignEventRepository, ForeignEventParticipantRepository.**
* **SquadParticipantRepository, ActParticipantRepository, ActParticipantBranchRepository.**

Сервисы – это разновидность классов, которые используются для интеграции различных систем или для реализации бизнес-логики.

В системе реализованы следующие сервисы:

* **DocumentOutService, DocumentInService** для обеспечения бизнес-логики документации.
* **OrderMainService, OrderEventService** и **OrderTrainingService** для обработки приказов.
* **PeopleService** для реализации бизнес-логики класса People.
* **RegulationService** для обеспечения бизнес-логики положений.
* **TeamService** для реализации бизнес-логики классов Team и TeamName.
* **OrderTrainingGroupParticipantService, TrainingGroupService** для реализации логики работы учебных групп.
* **ActParticipantService**, **ActParticipantBranchService, ForeignEventParticipantService**, **SquadParticipantService** и **ForeignEventService** для обеспечения логики мероприятий.

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 быть эффективными, чтобы обеспечивать точность обработки данных, а также удовлетворять потребности пользователя.

* 1. **Основные сценарии работы программного продукта**

Для отображения основных сценариев работы с системой используются диаграмма состояний и диаграмма активностей.

* + 1. **Диаграмма состояний**

В рассматриваемой системе существуют следующие сущности: документация (**Document**), учебная группа (**Training Group**) и ученик (**Student**). В Приложении 5представлены диаграммы состояний для трёх видов сущностей на стороне сервера.

Изначально в системе документ отсутствует, при его создании с помощью метода **actionCreate()**пребывает в активном состоянии, после чего на него может быть указан ответ с помощью метода **prepareUpdate()** класса **InOutDocumentCreateEvent,** 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 **prepareCreate()**. Документ может быть удалён с помощью метода **actionDelete().**

* + 1. **Диаграмма активности**

Сущности переходят из одного состояния в другое в следствие выполнения какой-либо деятельности. В Приложении 6представлены диаграммы активностей для документации.

В Приложении 6 на рисунке П6.1 представлена деятельность метода ***actionCreate()*** класса ***OrderEventController***, то есть создание объекта приказа о мероприятии на сервере (класс ***OrderEventWork)***. Тот в свою очередь не только создаёт приказ, но и определяет ответственных за его исполнение, создавая объекты класса ***OrderPeople****.* После добавления приказа образуются связи вида «мероприятие-команда-ученик-номинация». Затем через с помощью класса ***TeamEvent*** создаются команды и участники мероприятия, которые привязывается к мероприятию. В самом конце, как только все события были обработаны, информация о созданном приказе передаётся клиенту.

Входные данные:

* текстовые данные или файлы;
* выбор пунктов меню.

Выходные данные:

* сгенерированные документы в различных форматах;
* текстовые данные или файлы, полученные от пользователей.
  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.
* 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. **Безопасность ИС.**

Система ВКС должна соответствовать следующим требованиям безопасности:

* защита системы должна обеспечиваться комплексом программно-технических средств и поддерживающих их организационных мер;
* 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.

Общий уровень защищенности ИСПДн зависит от организации, использующей данный программный комплекс.

1. **РАБОЧИЙ 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  
      МОДЕЛИ:**

В таблице 2.1 приведено 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. |

**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. |

**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. |

**TrainingGroupParticipantWork:**

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

**OrderTrainingGroupParticipantWork:**

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

**ForeignEventWork:**

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

**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) |

**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 |

**OrderMainWork:**

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| scanFile | поле | Представляет собой сканированную копию приказа. |
| docFile | поле | Представляет собой отредактированную копию приказа. |
| appFIle | поле | Представляет собой файлы-приложения к приказу. |
| 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 приказа. |
| getFullNumber() | функция | Отображает полный номер приказа с постфиксом. |
| getFileLinks() | функция | Обеспечивает доступ к загруженным файлам приказа. |
| generateDocumentNumber() | функция | Генерирует номер приказа в системе. |

**OrderEventWork:**

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

**OrderTrainingWork:**

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| scanFile | поле | Представляет собой сканированную копию приказа. |
| docFile | поле | Представляет собой отредактированную копию приказа. |
| fill() | функция | Создает новый экземпляр класса с заданными параметрами. |
| getFullName() | функция | Отображает полное название приказа. |
| getFullNumber() | функция | Отображает полный номер приказа с постфиксом. |
| getFileLinks() | функция | Обеспечивает доступ к загруженным файлам приказа. |
| getNomenclature() | функция | Определяет статус номенклатуру приказа. |
| getStatus() | функция | Определяет статус приказа. |
| generateDocumentNumber() | функция | Генерирует номер приказа в системе. |

**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. |

**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. |

**SquadParticipantWork:**

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

**TeamNameWork:**

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

**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. |

**ActParticipantBranchWork:**

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

**OrderPeopleWork:**

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

**СЕРВИСЫ:**

**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. |

**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. |

**ForeignEventParticipantsService:**

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

**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 |

**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» |

**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» |

**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. |

**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. |

**OrderEventService:**

|  |  |  |
| --- | --- | --- |
| **Название элемента класса** | **Тип** | **Назначение** |
| \_\_construct() | функция | Создает экземпляр класса. |
| saveFilesFromModel() | функция | Сохраняет файла в каталогах системы и в БД. |

**OrderMainService:**

|  |  |  |
| --- | --- | --- |
| **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. |
| createOrderNumber() | функция | Создает номер приказа. |
| addOrderPeopleEvent() | функция | Создаёт связь «человек-ответственность» |
| deleteOrderPeopleEvent() | функция | Удаляет связь «человек-ответственность» |
| updateOrderPeopleEvent() | функция | Изменяет связь «человек-ответственность» |

**OrderTrainingService:**

|  |  |  |
| --- | --- | --- |
| **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. |
| isPossibleToInsertTrainingGroupParticipant() | функция | Проверяет возможность перевода/зачисления/отчисления |
| sPossibleToDeleteOrderTrainingGroupParticipant() | функция | Проверяет возможность отмены перевода/зачисления/отчисления |
| createOrderTrainingGroupParticipantEvent() | функция | Зачисляет/отчисляет/переводит ученика. |
| updateOrderTrainingGroupParticipantEvent() | функция | Изменяет положение ученика в рамках учебного процесса. |

**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. |

**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. |

**ActParticipantBranchService:**

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

**РЕПОЗИТОРИИ:**

**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. |

**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. |

**ForeignEventParticipantsRepository:**

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

**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 |

**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. |

**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. |

**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. |

**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. |

**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. |

**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» |

**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**

# **ЗАКЛЮЧЕНИЕ**

В процессе выполнения курсового проекта проектированию были решены следующие задачи:

* изучена предметная область;
* разработана различных диаграмм, иллюстрирующих 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 планирования 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 планирования образовательного процесса и документооборота в организации «Региональный школьный технопарк».

**Назначение разработки:** система предназначена для бесперебойной и эффективной работы системы автоматизации планирования 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.
* Эффективно добавлять и удалять файлы в системе.
* Проводить аналитику образовательного процесса.
* Изменять статусы учеников в системе.
  1. В системе не предусмотрен механизм личных сообщений и личного кабинета пользователя. **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**

**ФУНКЦИОНАЛЬНЫЕ ДИАГРАММЫ IDEF0**

Рисунок П2.1 - Функциональная диаграмма «Обработка документации»

Рисунок П2.2 - Функциональная диаграмма «Приказы о мероприятии»

**ПРИЛОЖЕНИЕ 3**

**ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ**



Рисунок П3.1 - Диаграмма вариантов использования

**ПРИЛОЖЕНИЕ 4**

**ДИАГРАММА КЛАССОВ**



Рисунок П4.1 – Диаграмма классов OrderMainWork, OrderMain, SearchOrderMain, SearchOrderEvent



Рисунок П4.2 – Диаграмма классов DocumentOutWork, DocumentOut, SearchDocumentOut



Рисунок П4.3 - Диаграмма классов DocumentInWork, DocumentIn, SearchDocumentIn



Рисунок П4.4 - Диаграмма классов ForeignEvent, ForeignEventWork



Рисунок П4.5 – Диаграмма классов ActParticipant, ActParticipantWork



Рисунок П4.6 – Диаграмма классов TeamWork, Team, TeamName, TeamNameWork



Рисунок П4.7 – Диаграмма классов Expire, ExpireWork, Regulation, RegulationWork



Рисунок П4.8 – Диаграмма классов People, PeopleWork, User, UserWork, Position, PositionWork



Рисунок П4.9 – Диаграмма классов Company, CompanyWork, Files, FilesWork



Рисунок П4.10 - Диаграмма связи классов DocumentInWork и DocumentOutWork



Рисунок П4.11 – Диаграмма связи классов People, Position, Company

**ПРИЛОЖЕНИЕ 5**

**ДИАГРАММА СОСТОЯНИЙ**



Рисунок П5.1 – Диаграмма сущности Document

**ПРИЛОЖЕНИЕ 6**

**ДИАГРАММА АКТИВНОСТИ**

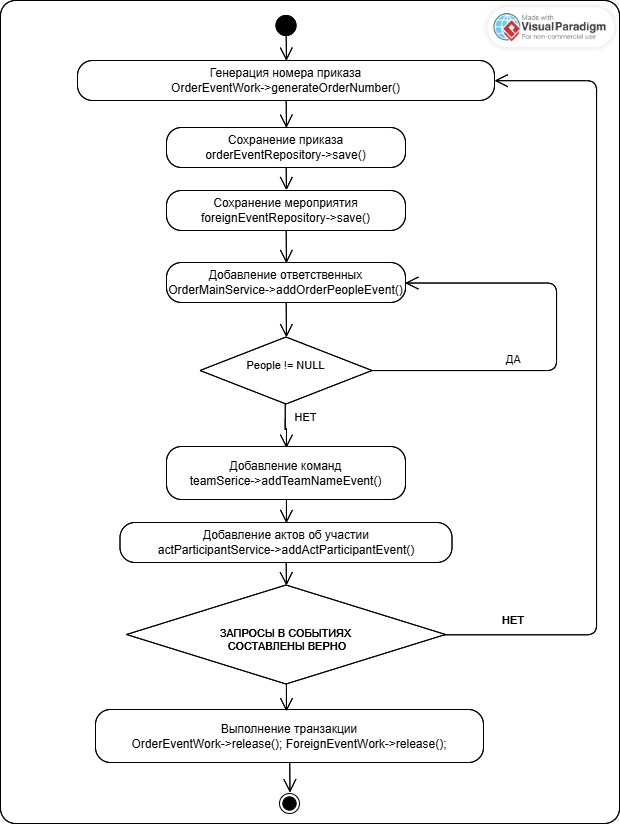


Рисунок П6.1 – Диаграмма активности «Приказа о мероприятии»