ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«БАШКИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Бирский филиал УУНиТ

Факультет физики и математики

Кафедра информатики и экономики

Направление подготовки:

09.03.03 Прикладная информатика

Направленность (профиль) образовательной программы:

Прикладная информатика в информационной сфере

Наименование дисциплины (модуля)

Проектный практикум

Курсовая работа

Информационная система «Цифровая галерея»

|  |  |
| --- | --- |
| Научный руководитель: | Выполнил: студент 4 курса  очной формы обучения  группы 4  Хусаинов Р. М. |
| канд. хим. наук, доцент кафедры  информатики и экономики |
| Мальцев Д. В. |
|  |

БИРСК – 2023**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc126143765)

[1. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc126143766)

[1.1. Исследование информационных процессов в нотации IDEF0, модель «Как есть» 5](#_Toc126143767)

[1.2. Формирование технического задания 8](#_Toc126143768)

[1.2.1. Общие сведения 8](#_Toc126143769)

[1.2.2. Назначение и цели создания системы 9](#_Toc126143770)

[1.2.3. Характеристика объектов автоматизации 10](#_Toc126143771)

[1.2.4. Требования к системе 11](#_Toc126143772)

[1.2.4.1. Требования к системе в целом 11](#_Toc126143773)

[1.2.4.1.1. Требования к структуре и функционированию системы 11](#_Toc126143774)

[1.2.4.1.2. Требования к численности и квалификации персонала системы и режиму его работы 13](#_Toc126143775)

[1.2.4.1.3. Требования к режимам работы персонала 13](#_Toc126143776)

[1.2.4.1.4. Требования к показателям назначения 14](#_Toc126143777)

[1.2.4.1.5. Требования к сохранению работоспособности системы в различных вероятных условиях 14](#_Toc126143778)

[1.2.4.1.6. Требования к надежности 16](#_Toc126143779)

[1.2.4.1.7. Требования к методам оценки и контроля показателей надежности на разных стадиях создания системы в соответствии с действующими нормативно-техническими документами 18](#_Toc126143780)

[1.2.4.2. Требования к функциям, выполняемым системой 20](#_Toc126143781)

[1.2.4.2.1. Подсистема хранения данных 20](#_Toc126143782)

[1.2.4.2.2. Подсистема формирования и визуализации изображений 22](#_Toc126143783)

[1.2.4.2.3. Подсистема авторизации администратора 24](#_Toc126143784)

[1.2.4.2.4. Подсистема управления контентом информационной системы 26](#_Toc126143785)

[1.3. Вывод 29](#_Toc126143786)

[2. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ 30](#_Toc126143787)

[2.1. Исследование на предмет автоматизации процесса IDEF0 (Как должно быть) 30](#_Toc126143788)

[2.2. Функционал системы 34](#_Toc126143789)

[2.3. Модели базы данных 39](#_Toc126143790)

[2.4. Выводы 42](#_Toc126143791)

[3. РЕАЛИЗАЦИЯ ПРОЕКТА 43](#_Toc126143792)

[3.1. Реализация front-end части информационной системы 43](#_Toc126143793)

[3.2. Реализация back-end части информационной системы 44](#_Toc126143794)

[3.3. Реализация базы данных 47](#_Toc126143795)

[ЗАКЛЮЧЕНИЕ 48](#_Toc126143796)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ. И ЛИТЕРАТУРЫ 49](#_Toc126143797)

[Приложение 1 51](#_Toc126143798)

[Приложение 2 55](#_Toc126143799)

[Приложение 3 61](#_Toc126143800)

ВВЕДЕНИЕ

Ещё 15–20 лет назад нельзя было предположить, что веб-приложения станут неотъемлемой частью жизни. Сегодня эта разновидность онлайн-инструментов используется для различных задач, включая оптимизацию бизнес-процессов, продажу товаров и услуг, распространение информации, общение пользователей друг с другом.

Веб-приложение представляет собой веб-сайт, на котором размещены страницы с частично либо полностью несформированным содержимым. Окончательное содержимое формируется только после того, как посетитель сайта запросит страницу с веб-сервера. В связи с тем, что окончательное содержимое страницы зависит от запроса, созданного на основе действий посетителя, такая страница называется динамической.

Любой информационный процесс может быть автоматизирован с помощью веб-приложения. Оно имеет неоспоримый ряд преимуществ: удобство развертывания, надежность, доступность, кроссплатформенность и функциональность. По этим причинам в данной работе оно было выбрано в качестве средства автоматизации.

Объект исследования – информационный процесс организации выставок произведений искусства.

Предмет исследования – автоматизация информационного процесса по организации выставок произведений искусств посредством реализации онлайн-выставок произведений искусств в виде веб-приложения.

Цель исследования – разработка прототипа централизованной системы хранения и визуализации изображений произведений искусств.

Задачи:

• Провести анализ предметной области;

• Спроектировать компоненты ИС;

• Разработать компоненты ИС;

1. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

## ****Исследование информационных процессов в нотации IDEF0, модель «Как есть»****

Исследуем информационные процессы предметной области в нотации IDEF0.

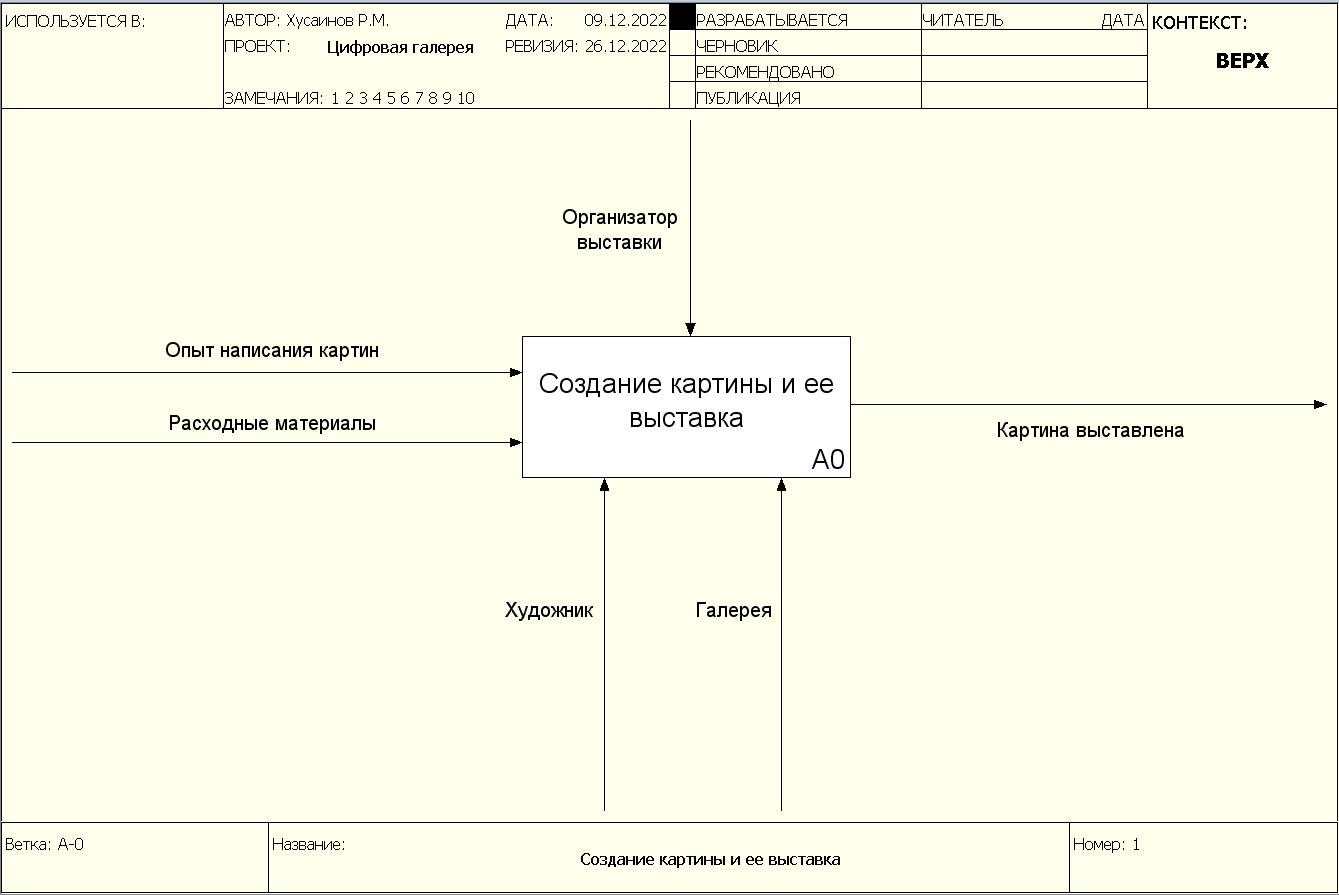


Рисунок 1. Контекстная диаграмма в нотации IDEF0, модель «Как есть», информационного процесса «Создание картины и ее выставка».

Нотация IDEF0 предназначена для формализации и описания бизнес – процессов. С помощью средств данной нотации была создана контекстная диаграмма (Рисунок 1.1), направленная на отражение процесса создания картины и организации ее выставки.

На представленной контекстной диаграмме можно выявить следующие типы стрелок:

Вход – имеющийся опыт написания картин и расходные материалы для создания картины. Они представляют собой объекты, которые будут использованы или преобразованы для получения результата (выхода);

Выход – выставка готовой картины в галерее. Объект, в который преобразуются входы;

Управление – организатор выставки. Представляет собой правила и требования, регламентирующие допуск картины к выставке.

Механизм – художник, создающий картину, и галерея, являющаяся местом выставки картины.

В настоящие время создание и выставка картины происходит следующим образом:

1. художник, руководствуясь опытом создания картин и имеющимися расходными материалами, формирует ключевую идею картины;
2. художник создает картину, используя свою сформировавшуюся идею;
3. готовая картина показывается организатору выставки. На этом этапе происходит рецензия картины.
4. в случае положительной оценки организатором, выставка картины согласуется с организатором, на данном этапе решаются организационные вопросы.

Расписанный процесс можно оформить следующей декомпозицией контекстной диаграммы, в которой были выявлены следующие функциональные блоки (Рисунок 1.2):

1. Формирование идеи картины
2. Создание картины
3. Рецензия картины
4. Согласование выставки

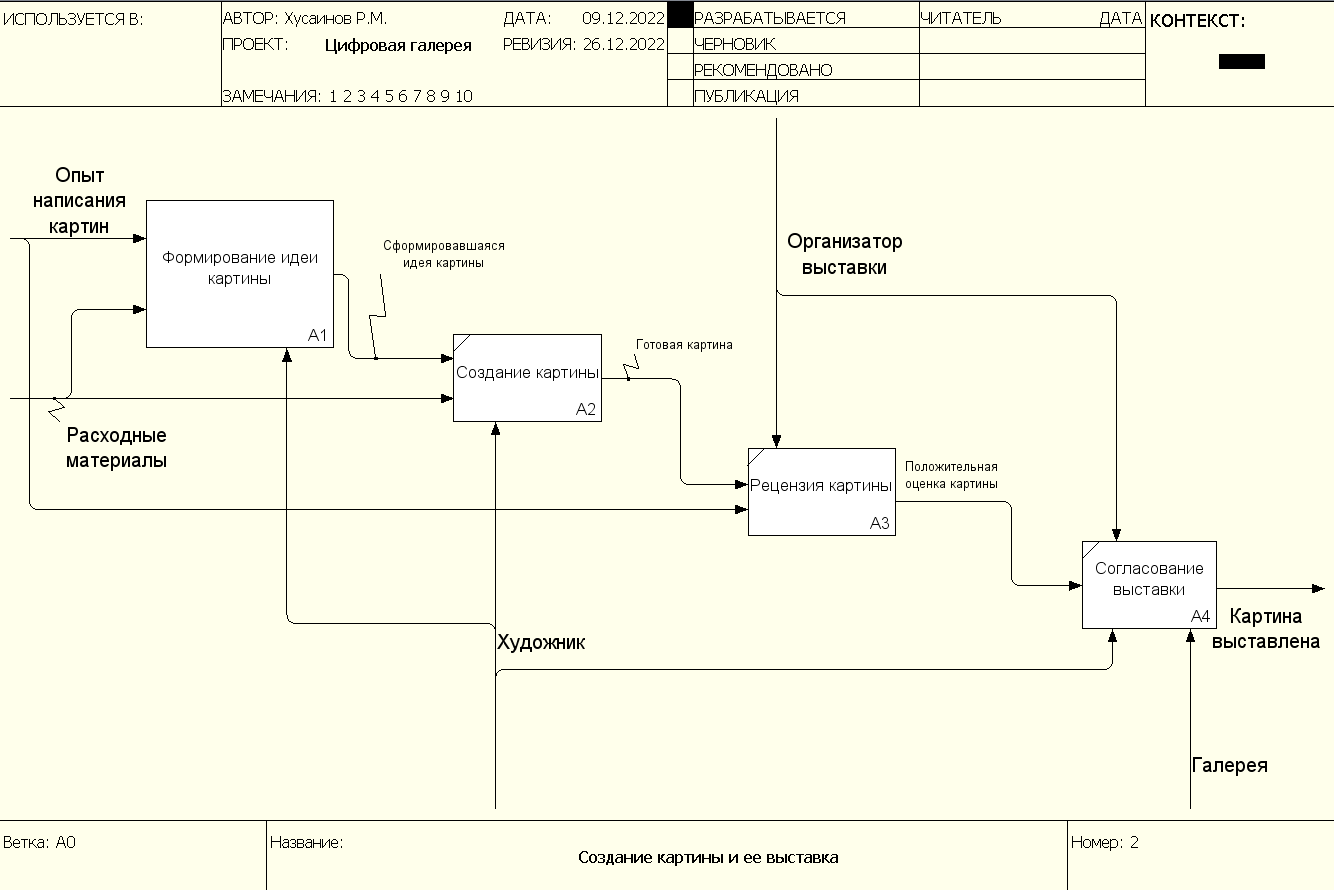


Рисунок 2. Результат декомпозиции контекстной диаграммы.

Данная декомпозиция (Рисунок 1.3) представляет собой иллюстрацию процесса формирования идеи картины. Формирование происходит в 2 этапа: рождение замысла и создание поисковых эскизов, этюдов для определения художественных параметров картины.

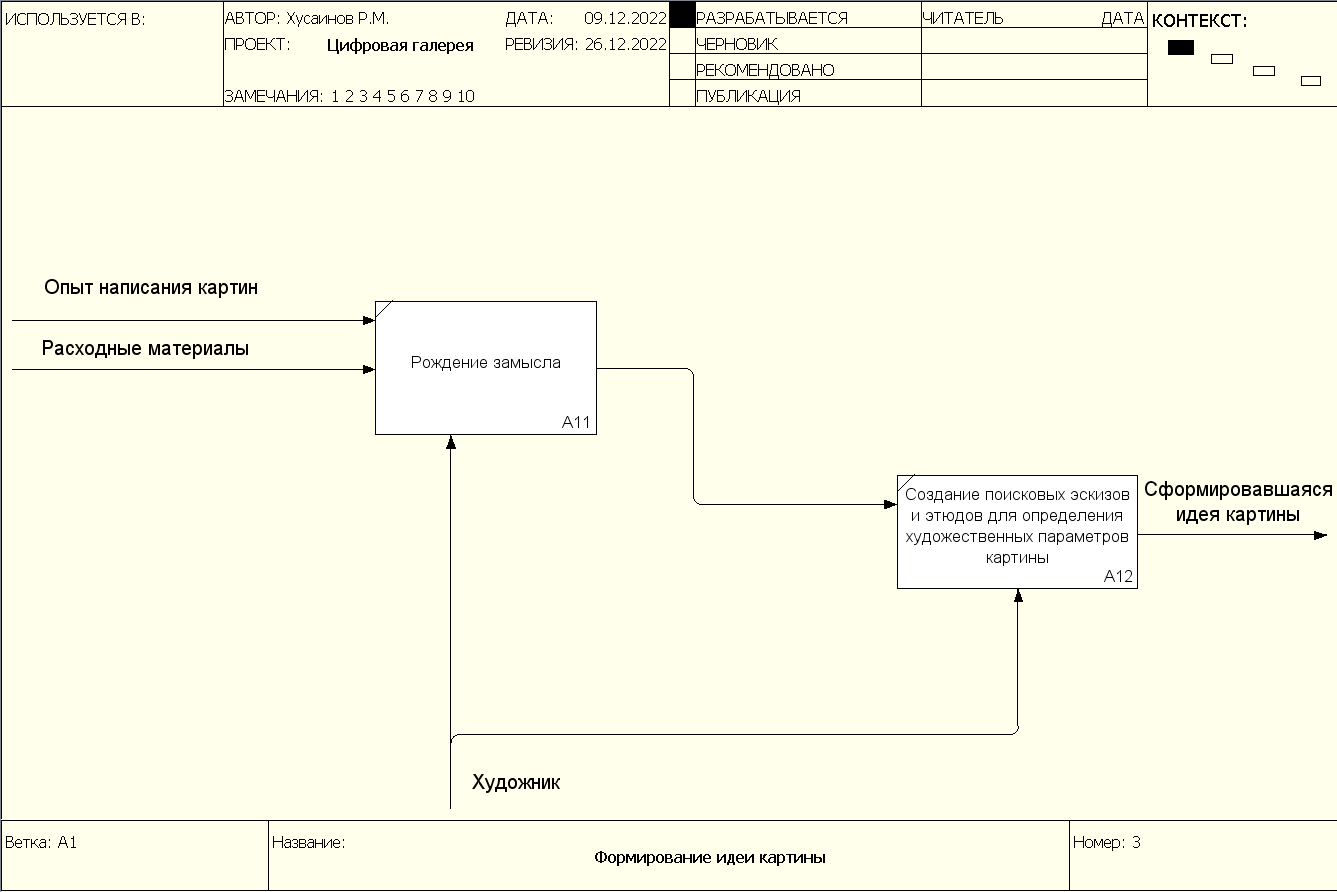


Рисунок 3. Диаграмма формирования идеи картины.

В ходе разработки структурно-функциональной диаграммы по типу «Как есть» были выявлены следующие недостатки:

- картины художника недоступны широкому кругу лиц;

- публикация картин только в офлайн галерее лишает художника шанса получить большее количество поклонников его творчества;

- в случае непреднамеренной порчи картины, ее нельзя выставить в оффлайн галерее.

## Формирование технического задания

## Общие сведения

Полное наименование системы: Информационная Система «Цифровая галерея».

Краткое наименование: «Цифровая галерея».

## Назначение и цели создания системы

Система предназначена для предоставления пользователю доступа к произведениям изобразительного искусства посредством веб-технологий.

В рамках проекта автоматизируется информационно-аналитическая деятельность в следующих бизнес-процессах

1. Хранение произведений изобразительного искусства в формате цифровых изображений;
2. Предоставление доступа к произведениям изобразительного искусства путем их публикации в информационной системе;
3. Организация онлайн-выставок, дублирующих выставки изобразительного искусства в офлайн-галереях;
4. Маркетинговая составляющая офлайн-галерей.

Предусматривается установка системы в пространствах, предназначенных для демонстрации изобразительного искусства.

Наименования и требуемые значения технических, технологических, производственно-экономических или других показателей объекта автоматизации, которые должны быть достигнуты в результате создания АИС; критерии оценки достижения целей создания системы.

«Цифровая галерея» создается с целью:

- обеспечения удобного доступа к произведениям галереи;

- создания единой системы хранения цифровых копий произведений изобразительного искусства, выставляющихся в галереях;

- предоставить возможность художникам осуществлять показ выполненных работ широкому кругу любителей искусства;

В результате создания информационной системы должны быть улучшены значения следующих показателей:

- посещаемость галереи;

- количество просмотров произведений изобразительного искусства галереи;

- известность галереи в широких кругах.

## Характеристика объектов автоматизации

Объектом автоматизации являются совокупность бизнес-процессов, выполняемых в галерее произведений изобразительного искусства.

Для удобства изложения и восприятия информации далее участником процессов автоматизации будет выступать не произведение изобразительного искусства в целом, а более конкретно – картина.

Главным объектом автоматизации является процесс показа выставляемых произведений искусства. Художник создает картину. Учреждение получает ее от художника и выставляет ее на показ во время одной из выставок.

Учреждение допускает посетителей к картинам либо свободно, либо по пропускам, билетам.

## Требования к системе

## Требования к системе в целом

## Требования к структуре и функционированию системы

Определяется перечень функциональных подсистем, их назначение и основные характеристики, требования к числу уровней иерархии и степени централизации системы.

Информационная система «Цифровая галерея» должна быть централизованной, т.е. все данные должны располагаться в центральном хранилище. Система должна иметь двухуровневую архитектуру:

1. Источник;

2. Хранилище

В Системе предлагается выделить следующие функциональные подсистемы:

- подсистема хранения данных;

- подсистема формирования и визуализации изображений;

- подсистема авторизации администратора;

- подсистема управления контентом информационной системы.

В качестве протокола взаимодействия между компонентами Системы на транспортно-сетевом уровне необходимо использовать протокол TCP/IP.

Для организации доступа пользователей к отчетности должен использоваться протокол презентационного уровня HTTP и его расширение HTTPS.

Источниками данных для Системы являются хранилища, содержащие изображения картин, полученных с помощью фотографических средств или графического редактора.

Система должна поддерживать следующие режимы функционирования:

- Режим пользователя, в котором все подсистемы, кроме подсистемы управления контентом, выполняют все свои основные функции;

- Режим администратора, в котором присутствует возможность использования подсистемы управления контентом.

- Режим профилактических работ.

В основном режиме функционирования Информационная система «Цифровая галерея» должна обеспечивать:

- работу пользователей в режиме – 24 часов в день, 7 дней в неделю (24х7);

- выполнение своих функций – просмотр страниц веб-сайта с изображениями картин;

-хранение данных.

В режиме администратора она должна обеспечивать возможность проведения следующих работ:

- загрузка изображений картин, удаление, обновление и просмотр загруженных изображений;

- создание онлайн-выставки.

В режиме профилактических работ должна быть предусмотрена возможность предупреждения пользователей системы о недоступности системы в данный момент.

Для обеспечения высокой надежности функционирования Системы как системы в целом, так и её отдельных компонентов должно обеспечиваться выполнение требований по диагностированию ее состояния.

Диагностирование Системы должно осуществляться следующими штатными средствами:

- СУБД - SQLite

- Браузер с инструментами отладки DevTools – Google Chrome

- Отдельный HTTP-клиент - Postman

Для всех технических компонентов необходимо обеспечить регулярный и постоянный контроль состояния и техническое обслуживание, что должно обеспечиваться организацией, предоставляющей сервер для развертывания Информационной Системы.

## Требования к численности и квалификации персонала системы и режиму его работы

В состав персонала, необходимого для обеспечения эксплуатации «Цифровой галереи» в рамках соответствующих подразделений Заказчика, необходимо выделение следующих ответственных лиц:

- Администратор подсистем хранения данных, формирования и визуализации изображений, авторизации администратора. На всем протяжении функционирования ИС обеспечивает распределение дискового пространства, модификацию структур БД, оптимизацию производительности

- Администратор подсистемы управления контентом. Занимается обеспечением подготовки и загрузки данных из внешних источников в хранилище данных.

К квалификации персонала, эксплуатирующего Систему «Цифровая галерея», предъявляются следующие требования:

- Администратор подсистемы управления контентом - знание соответствующей предметной области, непосредственное отношение к учреждению, использующему данную ИС, а также навыки работы с CMS системами.

- Администратор подсистем хранения данных, формирования и визуализации изображений, авторизации администратора. Требуется знание и опыт администрирования СУБД, знание языка запросов SQL, владение навыками работы с реляционной базой данных SQLite.

## Требования к режимам работы персонала

Персонал, работающий с Системой «Цифровая галерея» и выполняющий функции её сопровождения и обслуживания, должен работать в следующих режимах:

- Администратор подсистемы управления контентом - знание соответствующей предметной области, непосредственное отношение к учреждению, использующему данную ИС, а также навыки работы с CMS системами.

- Администратор подсистем хранения данных, формирования и визуализации изображений, авторизации администратора – двухсменный график, поочередно.

## Требования к показателям назначения

Система должна обеспечивать следующие количественные показатели, которые характеризуют степень соответствия ее назначению:

- Количество пользователь, одновременно использующих ИС ~ 1000;

- Максимальное количество загружаемых изображений ~ 1000000;

Обеспечение приспособляемости системы должно выполняться за счет:

- своевременности администрирования;

- модернизации процессов сбора, обработки и загрузки данных в соответствии с новыми требованиями;

- модификации процедур доступа и представления данных конечным пользователям;

- наличия настроечных и конфигурационных файлов у ПО подсистем;

## Требования к сохранению работоспособности системы в различных вероятных условиях

В зависимости от различных вероятных условий система должна выполнять требования, приведенные в таблице.

Таблица 1

Вероятные условия

|  |  |
| --- | --- |
| Вероятное условие | Требование |
| Нарушения в работе системы внешнего электроснабжения серверного оборудования продолжительностью до 15 мин. | Функционирование в полном объеме. |
| Выход из строя сервера подсистемы хранения данных | Уведомление администратора подсистемы хранения данных |

## Требования к надежности

Уровень надежности должен достигаться согласованным применением организационных, организационно-технических мероприятий и программно-аппаратных средств.

Надежность должна обеспечиваться за счет:

- применения технических средств, системного и базового программного обеспечения, соответствующих классу решаемых задач;

- своевременного выполнения процессов администрирования Системы «Цифровая галерея»;

- соблюдения правил эксплуатации и технического обслуживания программно-аппаратных средств;

- предварительного обучения обслуживающего персонала.

Время устранения отказа должно быть следующим:

- при перерыве и выходе за установленные пределы параметров электропитания - не более 1 сутки.

- при перерыве и выходе за установленные пределы параметров программного обеспечением - не более 6 часов.

Система должна соответствовать следующим параметрам:

- среднее время восстановления 3 часа - определяется как сумма всех времен восстановления за заданный календарный период, поделенные на продолжительность этого периода;

При работе системы возможны следующие аварийные ситуации, которые влияют на надежность работы системы:

- сбой в электроснабжении сервера;

- сбой в электроснабжении у пользователей системы;

- сбой в электроснабжении обеспечения локальной сети (поломка сети);

- ошибки Системы «Цифровая галерея», не выявленные при отладке и испытании системы;

- сбои программного обеспечения сервера.

К надежности оборудования предъявляются следующие требования:

- в качестве аппаратных платформ должны использоваться средства с повышенной надежностью;

- применение технических средств соответствующих классу решаемых задач;

- аппаратно-программный комплекс Системы должен иметь возможность восстановления в случаях сбоев.

К надежности электроснабжения предъявляются следующие требования:

- с целью повышения отказоустойчивости системы в целом необходима обязательная комплектация серверов источником бесперебойного питания с возможностью автономной работы системы не менее 2 часов;

- система должны быть укомплектована подсистемой оповещения Администраторов о переходе на автономный режим работы;

- должно быть обеспечено бесперебойное питание активного сетевого оборудования.

Надежность аппаратных и программных средств должна обеспечиваться за счет следующих организационных мероприятий:

- предварительного обучения обслуживающего персонала;

- своевременного выполнения процессов администрирования;

- соблюдения правил эксплуатации и технического обслуживания программно-аппаратных средств;

- своевременное выполнение процедур резервного копирования данных.

Надежность программного обеспечения подсистем должна обеспечиваться за счет:

- надежности общесистемного ПО и ПО, разрабатываемого Разработчиком;

- проведением комплекса мероприятий отладки, поиска и исключения ошибок.

- ведением журналов системных сообщений и ошибок по подсистемам для последующего анализа и изменения конфигурации.

## Требования к методам оценки и контроля показателей надежности на разных стадиях создания системы в соответствии с действующими нормативно-техническими документами

Проверка выполнения требований по надежности должна производиться на этапе проектирования расчетным путем, а на этапах испытаний и эксплуатации - по методике Разработчика, согласованной с Заказчиком.

* + - * 1. **Требования к эргономике и технической эстетике**

Подсистема авторизации должна быть представлена в виде веб-страницы с интуитивным и понятным интерфейсом.

Внешнее оформление подсистемы авторизации должно отвечать следующим требованиям:

- интерфейс подсистемы должен быть выполнен в стиле остальных подсистем;

- должно быть обеспечено наличие локализованного (русскоязычного) интерфейса пользователя;

- должен использоваться шрифт: Roboto

- размер шрифта должен быть: 14

- цветовая палитра должна состоять из следующих цветов: #FFFFFF, #009933,

К подсистеме управления контентом предъявляются следующие требования к эргономике и технической эстетике.

В части внешнего оформления:

- интерфейс подсистемы должен быть выполнен в стиле остальных подсистем;

- должно быть обеспечено наличие локализованного (русскоязычного) интерфейса пользователя;

- должен использоваться шрифт: Roboto

- размер шрифта должен быть: 14

- интерфейс должен быть понятным, функциональным и простым.

В части диалога с пользователем:

- при возникновении ошибок в работе подсистемы на экран монитора должно выводиться сообщение с наименованием ошибки и с рекомендациями по её устранению на русском языке.

К подсистеме формирования и визуализации изображений предъявляются следующие требования к эргономике и технической эстетике.

В части внешнего оформления:

- изображения должны быть представлены в масштабе, комфортном для просмотра. Для этого требуется предусмотреть возможность увеличения изображения на весь экран.

- для уменьшения нагрузки на сеть пользователя требуется уменьшать размер изображения для его отображения в качестве миниатюры.

## Требования к функциям, выполняемым системой

## Подсистема хранения данных

Таблица 2

Функции подсистемы хранения данных

|  |  |
| --- | --- |
| Функция | Задача |
| Управляет процессами сбора, обработки и загрузки данных | Создание, редактирование и удаление процессов сбора, обработки и загрузки данных |
| Формирование последовательности выполнения процессов сбора, обработки и загрузки данных |
| Протоколирует результаты сбора, обработки и загрузки данных | Ведение журналов результатов сбора, обработки и загрузки данных |

Таблица 3

Временной регламент реализации каждой функции, задачи

|  |  |
| --- | --- |
| Задача | Требования к временному регламенту |
| Создание, редактирование и удаление процессов сбора, обработки и загрузки данных | Весь период функционирования системы, при возникновении необходимости изменения процессов сбора, обработки и загрузки данных |
| Формирование последовательности выполнения процессов сбора, обработки и загрузки данных | Весь период функционирования системы, при возникновении необходимости модификации регламента загрузки данных |
| Ведение журналов результатов сбора, обработки и загрузки данных | Регулярно, при работе подсистемы |

Таблица 4

Требования к качеству реализации функций, задач

|  |  |  |
| --- | --- | --- |
| Задача | Форма представления выходной информации | Характеристики точности и времени выполнения |
| Создание, редактирование и удаление процессов сбора, обработки и загрузки данных | Данные в структурах БД, которые представляются в компонентах веб-сайта | Определяется регламентом эксплуатации |
| Формирование последовательности выполнения процессов сбора, обработки и загрузки данных | Данные в структурах БД, которые представляются в компонентах веб-сайта | Определяется регламентом эксплуатации |
| Ведение журналов результатов сбора, обработки и загрузки данных | Текстовые файлы | В момент выполнения сбора, обработки и загрузки данных |

Таблица 5

Перечень критериев отказа для каждой функции

|  |  |  |  |
| --- | --- | --- | --- |
| Функция | Критерии отказа | Время восстановления | Коэффициент готовности |
| Управляет процессами сбора, обработки и загрузки данных | Не выполняется одна из задач функции. | 8 часов | 0.85 |
| Протоколирует результаты сбора, обработки и загрузки данных | Не выполняется одна из задач функции. | 12 часов | 0.75 |

## Подсистема формирования и визуализации изображений

Таблица 6

Перечень функций, задач подлежащих автоматизации

|  |  |
| --- | --- |
| Функция | Задача |
| Формирует изображения картин, оптимизированных для использования в веб-сайтах | Обеспечение веб-сайта изображениями картин оптимального размера |
| Для полноэкранного просмотра подгружает изображение высокого разрешения | Обеспечение пользователя возможностью просмотра изображений картин в режиме полноэкранного просмотра |

Таблица 7

Временной регламент реализации каждой функции, задач

|  |  |
| --- | --- |
| Задача | Требования к временному регламенту |
| Обеспечение веб-сайта изображениями картин оптимального размера | Реализация до начала функционирования системы |
| Обеспечение пользователя возможностью просмотра изображений картин в режиме полноэкранного просмотра | Реализация до начала функционирования системы |

Таблица 8

Требования к качеству реализации функций, задач

|  |  |  |
| --- | --- | --- |
| Задача | Форма представления выходной информации | Характеристики точности и времени выполнения |
| Обеспечение веб-сайта изображениями картин оптимального размера данных | Изображения с расширениями .jpeg, .png со сжатием | Определяется регламентом эксплуатации |
| Обеспечение пользователя возможностью просмотра изображений картин в режиме полноэкранного просмотра | Изображения с расширениями .jpeg, .png без сжатия | Определяется регламентом эксплуатации |

Таблица 9

Перечень критериев отказа для каждой функции

|  |  |  |  |
| --- | --- | --- | --- |
| Функция | Критерии отказа | Время восстановления | Коэффициент готовности |
| Формирует изображения картин, оптимизированных для использования в веб-сайтах | Не выполняется задача функции. | 2 часа | 0.85 |
| Для полноэкранного просмотра подгружает изображение высокого разрешения | Не выполняется задача функции. | 2 часа | 0.75 |

## Подсистема авторизации администратора

Таблица 10

Перечень функций, задач подлежащих автоматизации

|  |  |
| --- | --- |
| Функция | Задача |
| Обеспечивает доступ Администратора к системе | Вход в систему через специальную страницу авторизации |
| Обеспечивает защиту от злоумышленников | Защита системы от неправомерного доступа |

Таблица 11

Временной регламент реализации каждой функции, задачи

|  |  |
| --- | --- |
| Задача | Требования к временному регламенту |
| Вход в систему через специальную страницу авторизации | Реализация до начала функционирования системы |
| Защита системы от неправомерного доступа | Реализация до начала функционирования системы |

Таблица 12

Требования к качеству реализации функций, задач

|  |  |  |
| --- | --- | --- |
| Задача | Форма представления выходной информации | Характеристики точности и времени выполнения |
| Вход в систему через специальную страницу авторизации | При успешной авторизации происходит загрузка веб-страницы системы управления контентом. | Определяется регламентом эксплуатации |
| Защита системы от неправомерного доступа | Текстовый документ, содержащий информацию о попытках входа в систему | Определяется регламентом эксплуатации |

Таблица 13

Перечень критериев отказа для каждой функции

|  |  |  |  |
| --- | --- | --- | --- |
| Функция | Критерии отказа | Время восстановления | Коэффициент готовности |
| Обеспечивает доступ Администратора к системе | Не выполняется задача функции. | 2 часа | 0.85 |
| Обеспечивает защиту от злоумышленников | Не выполняется задача функции. | 2 часа | 0.75 |

## Подсистема управления контентом информационной системы

Таблица 14

Перечень функций, задач подлежащих автоматизации

|  |  |
| --- | --- |
| Функция | Задача |
| Формирует интерфейс для работы с данными (изображениями, художниками) | Обеспечение администратора подсистемой для работы с данными в информационной системе «Цифровая галерея» |
| Внесение изменений в базу данных | Добавление/чтение/обновление/удаление данных в БД без необходимости написания запросов к БД |

Таблица 15

Временной регламент реализации каждой функции, задачи

|  |  |
| --- | --- |
| Задача | Требования к временному регламенту |
| Обеспечение администратора подсистемой для работы с данными в информационной системе «Цифровая галерея» | Реализация до начала функционирования системы |
| Добавление/чтение/обновление/удаление данных в БД без необходимости написания запросов к БД | Реализация до начала функционирования системы |

Таблица 16

Требования к качеству реализации функций, задач

|  |  |  |
| --- | --- | --- |
| Задача | Форма представления выходной информации | Характеристики точности и времени выполнения |
| Обеспечение администратора подсистемой для работы с данными в информационной системе «Цифровая галерея» | Веб-приложение, позволяющее производить CRUD операции над сущностями БД | Реализация до начала функционирования системы |
| Добавление, чтение, обновление, удаление данных в БД без необходимости написания запросов к БД | Запросы к БД, сформированные с помощью подсистемы | Реализация до начала функционирования системы |

Таблица 17

Перечень критериев отказа для каждой функции

|  |  |  |  |
| --- | --- | --- | --- |
| Функция | Критерии отказа | Время восстановления | Коэффициент готовности |
| Формирует интерфейс для работы с данными (изображений, художников) | Не выполняется задача функции. | 2 часа | 0.85 |
| Внесение изменений в базу данных | Не выполняется задача функции. | 2 часа | 0.75 |

## Вывод

В данной главе была исследована предметная область, были изучены ее бизнес-процессы. В результате данного анализа было сформировано техническое задание для разработки информационной системы.

1. Проектирование Информационной Системы

## Исследование на предмет автоматизации процесса IDEF0 (Как должно быть)

По ходу разработки бизнес-процесса была построена структурно-функциональная диаграмма «Как должно быть?». Был добавлен новый ресурс – информационная система (Рисунок 2.1).

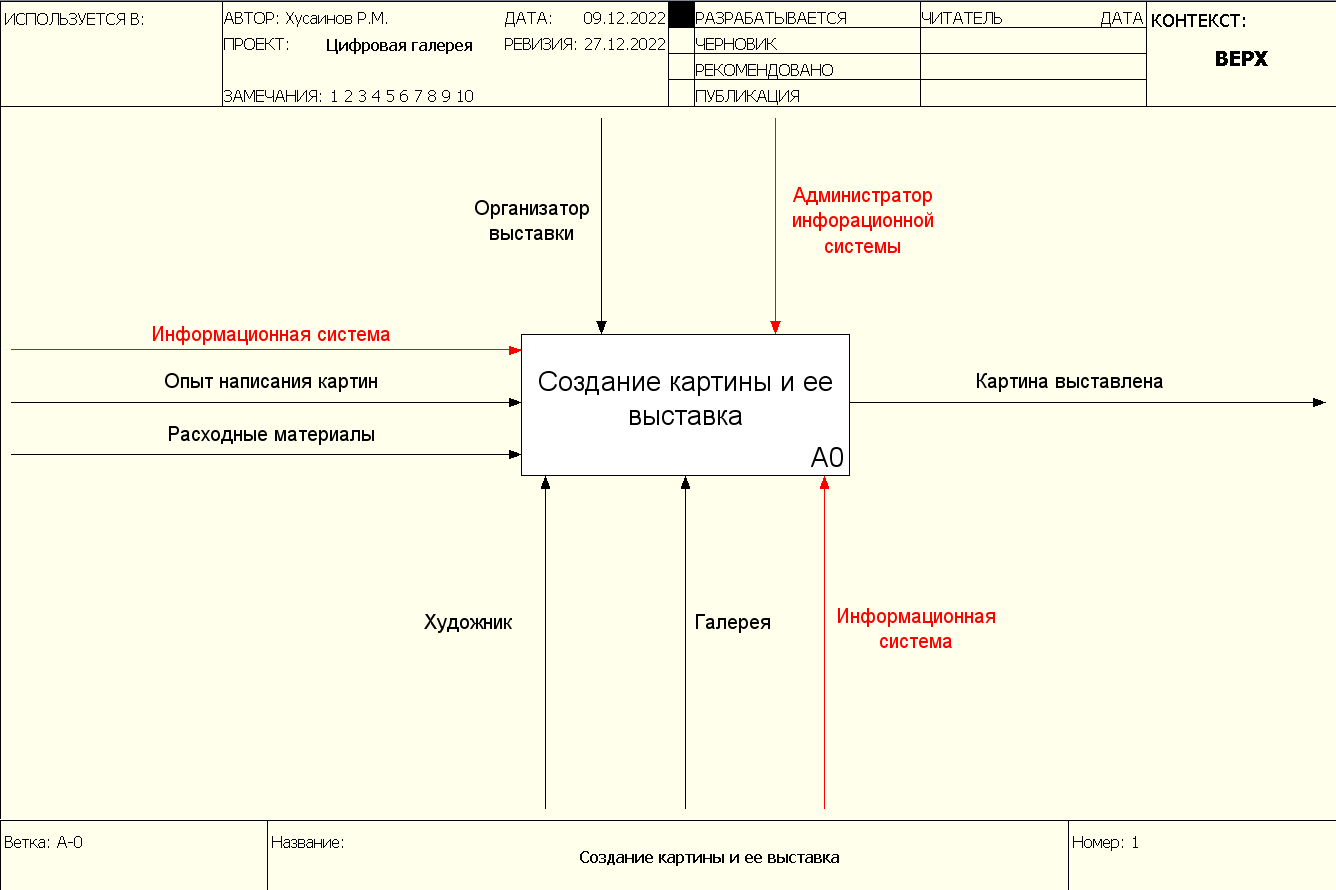


Рисунок 4. Контекстная диаграмма информационного процесса “Создание картины и ее выставка”, модель "Как должно быть?".

Декомпозиция контекстной диаграммы включает в себя следующие функциональные блоки (Рисунок 2.2):

1. Формирование идеи картины
2. Создание картины
3. Рецензия картины
4. Согласование выставки

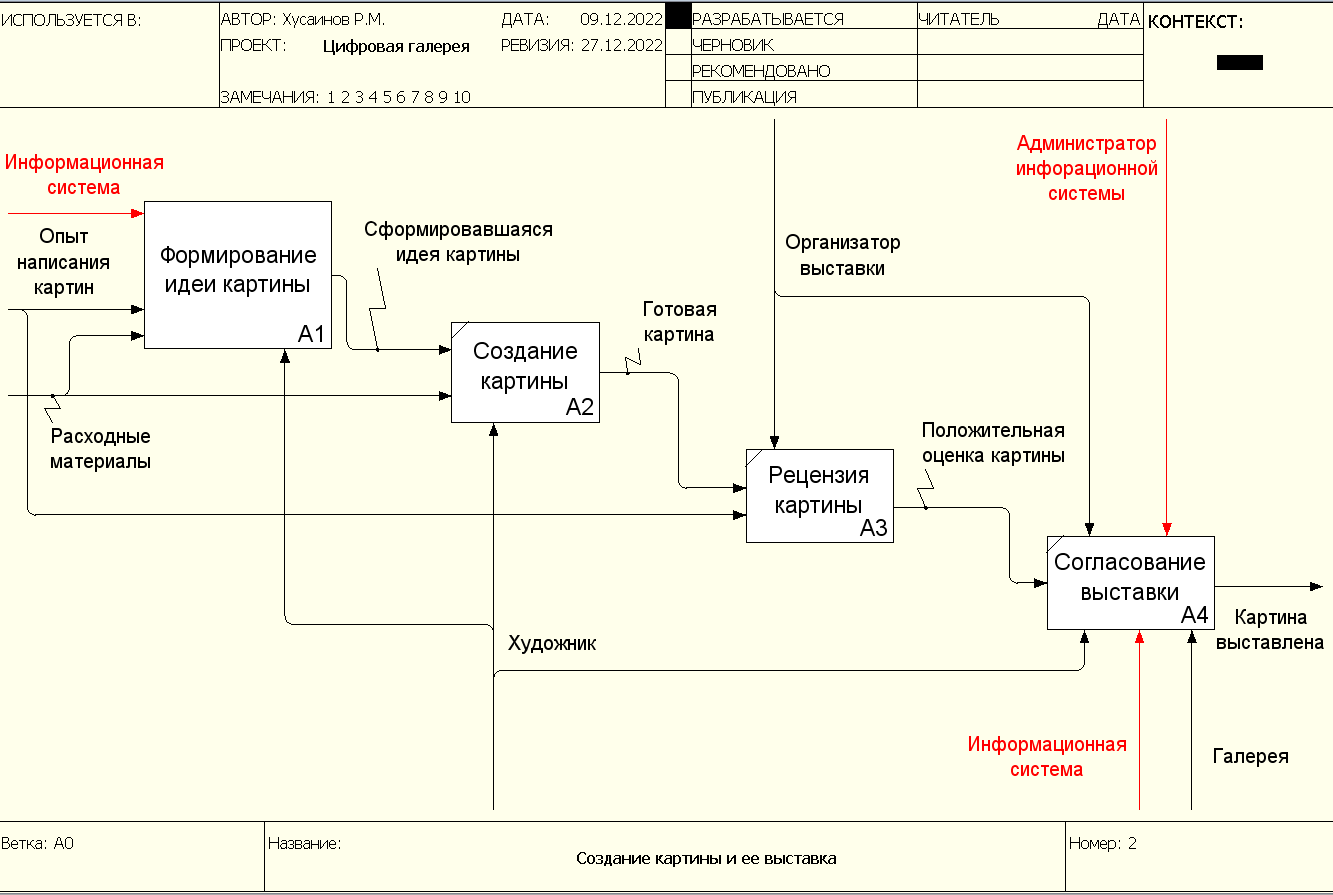


Рисунок 5. Результат декомпозиции контекстной диаграммы "Как должно быть?".

На основании анализа существующей процедуры создания и публикации картины, можно сделать вывод о необходимости ее улучшения путем автоматизации этого процесса. Применение специализированной информационной системы нацелено на решение данной задачи.

На рисунке 2.3 изображена декомпозиция функционального блока «Формирование идеи картины». Информационная система в данном случае выступает источником вдохновения художника, она обеспечивает его примерами ранее выполненных художественных работ, произведенных им самим, либо же другими художниками, выставляющими свои картины в данной информационной системе – онлайн-галерее.

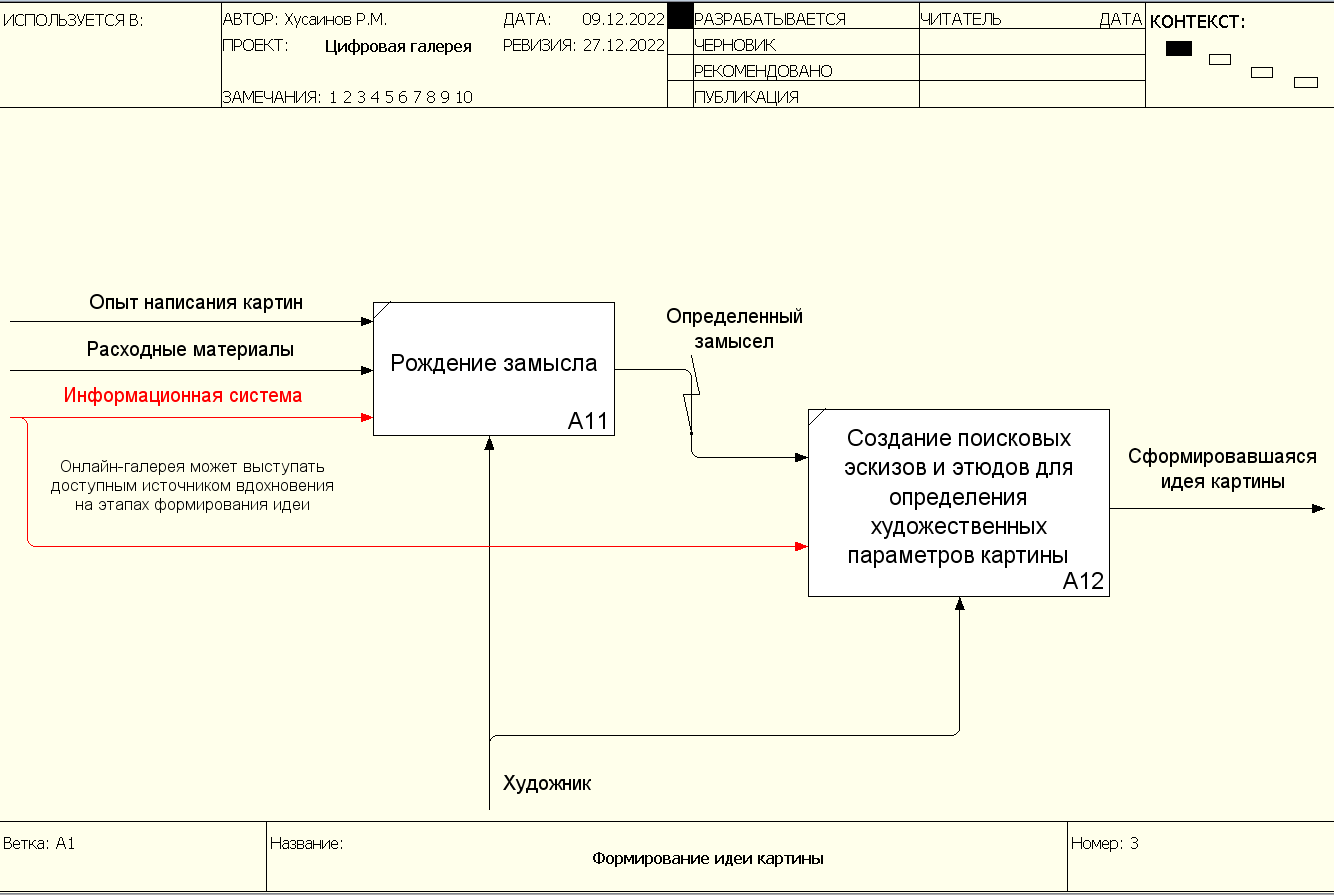


Рисунок 6. Результат декомпозиции функционального блока “Формирование идеи”, модель “Как должно быть”.

Таким образом, после внедрения информационной системы рассматриваемая операция будет выполняться по следующим этапам:

1. Художник формирует идею, пользуясь всеми доступными для него средствами. В контексте данной модели теперь ему доступна онлайн-галерея.
2. Художник создает картину на основе сформированной ранее идее, пользуясь необходимыми расходными материалами. Конечный результат – передача картины на этап рецензии.
3. Организатор выставки оценивает нарисованную художником картину. В случае положительной оценки, картина допускается для выставки.
4. На этапе согласования выставки ее организатор и автор картины договариваются о времени и месте выставки. В контексте модели «Как должно быть» в качестве места выставки теперь так же присутствует Информационная система «Цифровая галерея», выступая в роли онлайн-галереи для публикации произведения. В процессе публикации картины участвует Администратор Информационной системы, который имеет доступ к системе и публикует картину. Таким образом, картина выставляется на двух площадках, одна из которых онлайн.

Описанный процесс представлен функциональной диаграммой «Как должно быть», изображенной на рисунке 2.2.

Он позволяет получить следующие преимущества:

1. Созданная художником картина одновременно может публиковаться сразу на двух площадках. Первой площадкой является офлайн-галерея, например, галерея Бирского филиала УУНиТ. Второй же является онлайн-галерея – данный проект.
2. Картина, опубликованная в информационной системе, доступна всем пользователям сети интернет. Это обеспечивает художнику необходимую обратную связь, позволяет обрести поклонников творчества, геолокация которых может быть абсолютно разной.
3. Картину можно выставить в онлайн-галерее, даже если ее подлинник утрачен. Таким образом, даже в цифровом виде картина позволяет быть увиденной благодаря внедрению данной информационной системы.

Описание выполняемых функций.

В состав информационной системы входят следующие функции:

1. Панель управления системой для администратора;
2. Авторизация для администратора;
3. Смена логина и пароля администратора;
4. Добавление, удаление картин через панель управления системой;
5. Хранение картин;
6. Возможность получения доступа к цифровым копиям картин высокого разрешения.

## Функционал системы

В соответствии с техническим заданием были выявлены основные прецеденты для описания системы на концептуальном уровне и составлены соответствующие диаграммы последовательностей, позволяющие наглядно показать взаимодействие акторов в информационной системе.

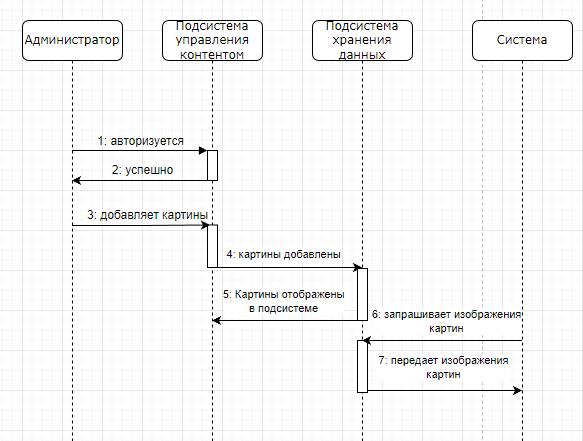


Рисунок 7. Диаграмма последовательностей для прецедента управления данными на примере, добавления картины в базу данных посредством системы управления контентом.



Рисунок 8. Диаграмма последовательностей для прецедента использования поиска пользователем.

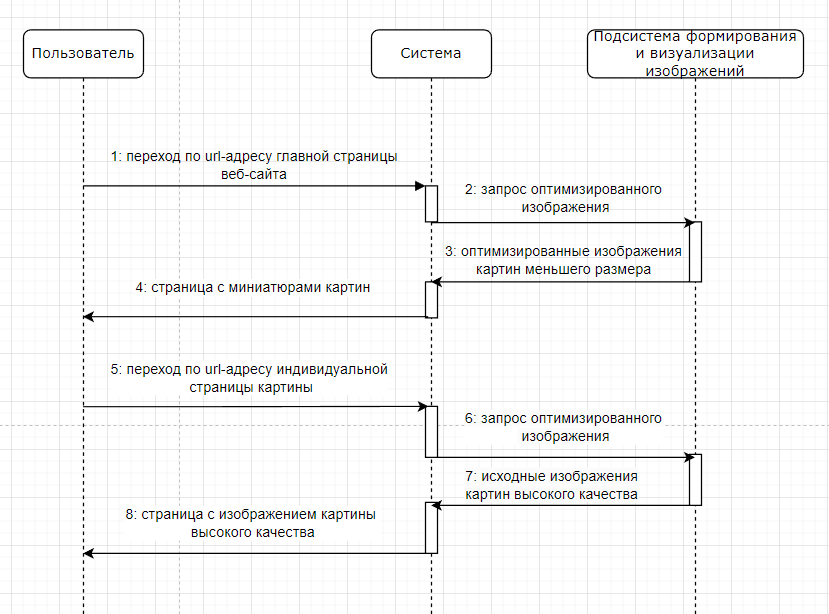


Рисунок 9. Диаграмма последовательностей для прецедента использования подсистемы формирования изображений.

Определены состояния для компонентов системы.

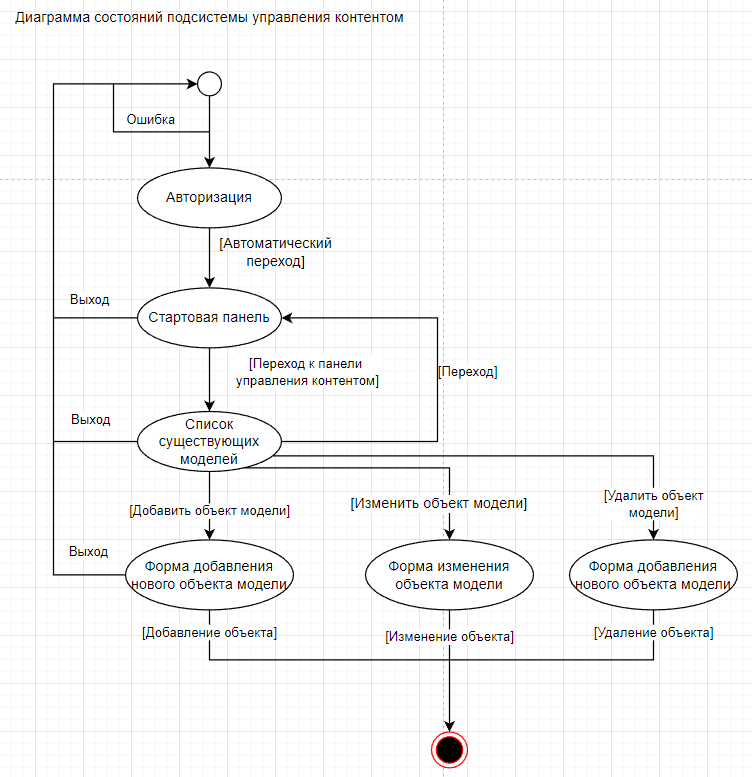


Рисунок 10. Диаграмма состояний подсистемы управления контентом.

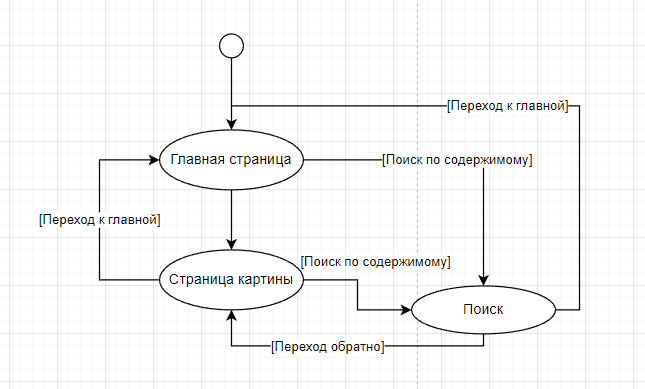


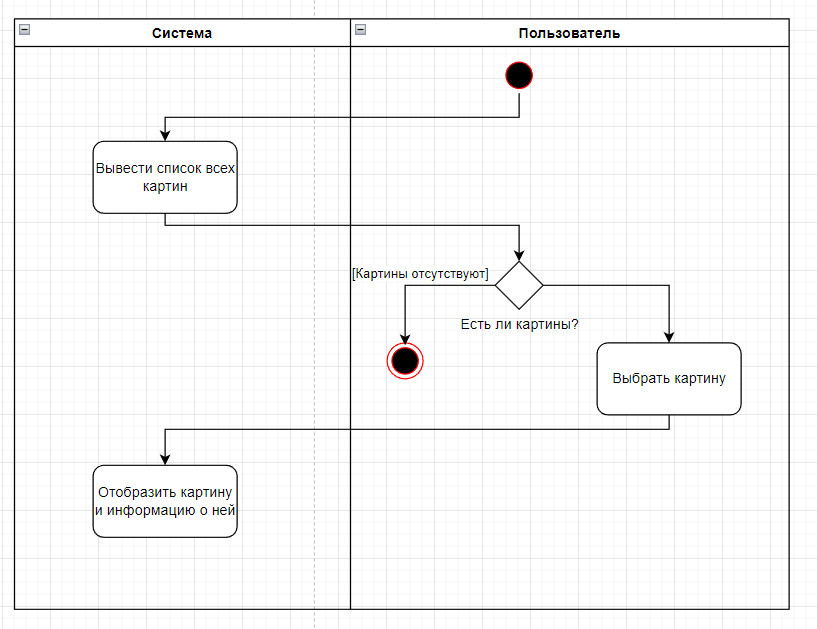
Рисунок 11. Диаграмма состояний системы при использовании ее пользователем.

Рисунок 12. Диаграмма деятельности, визуализирующая случай использования системы пользователем.

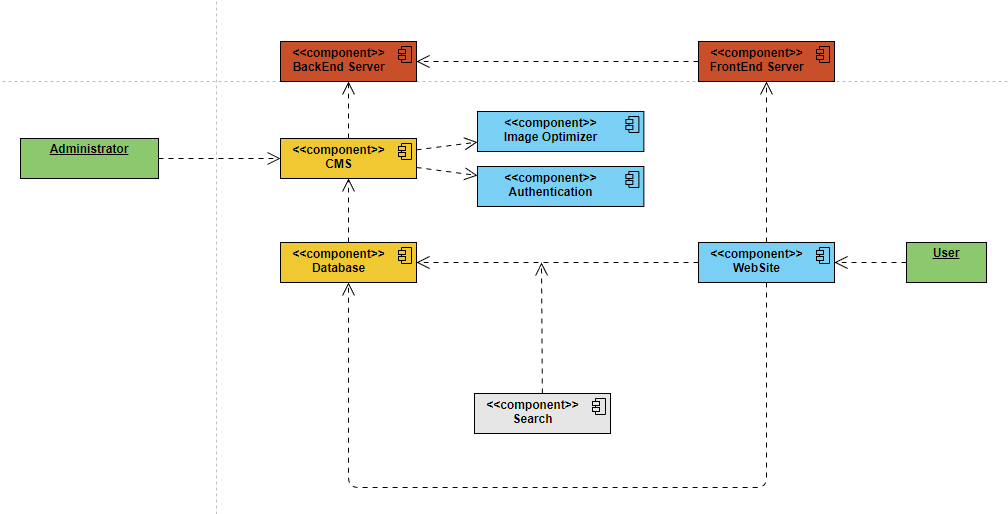


Рисунок 13. Диаграмма компонентов системы, описывающая физическое размещение системы и расположение ее отдельных подсистем.

## Модели базы данных

Сформированы инфологическая и даталогическая модели БД.

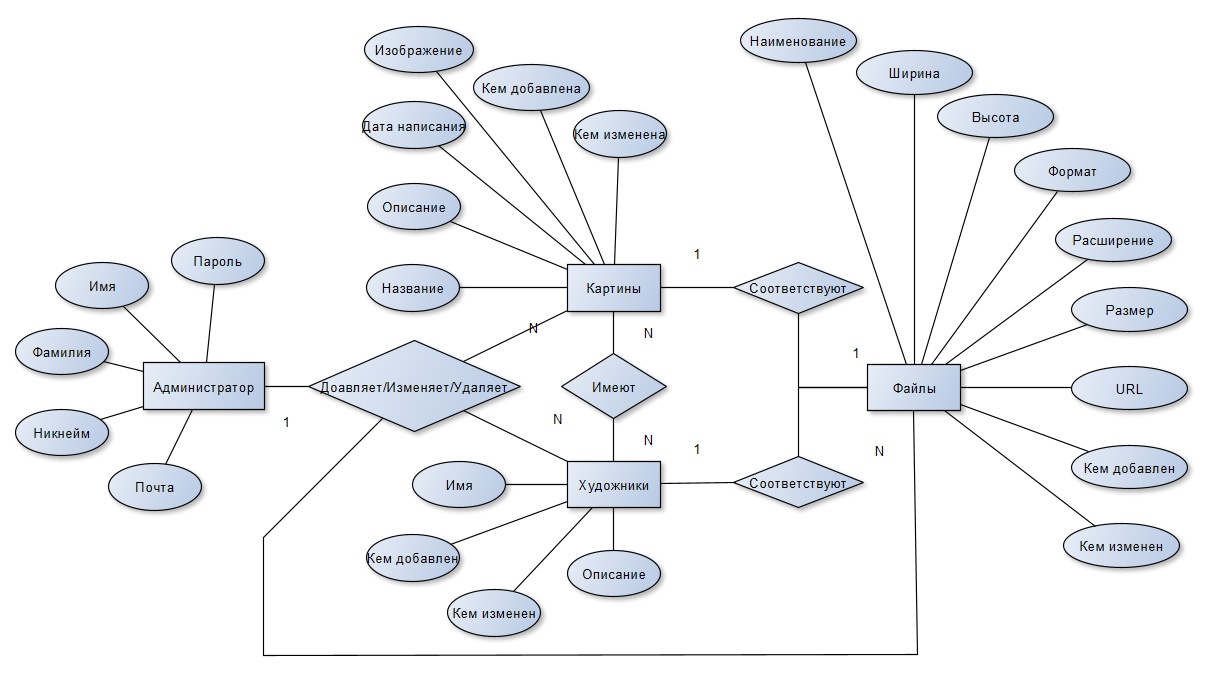


Рисунок 15. Инфологическая модель БД.

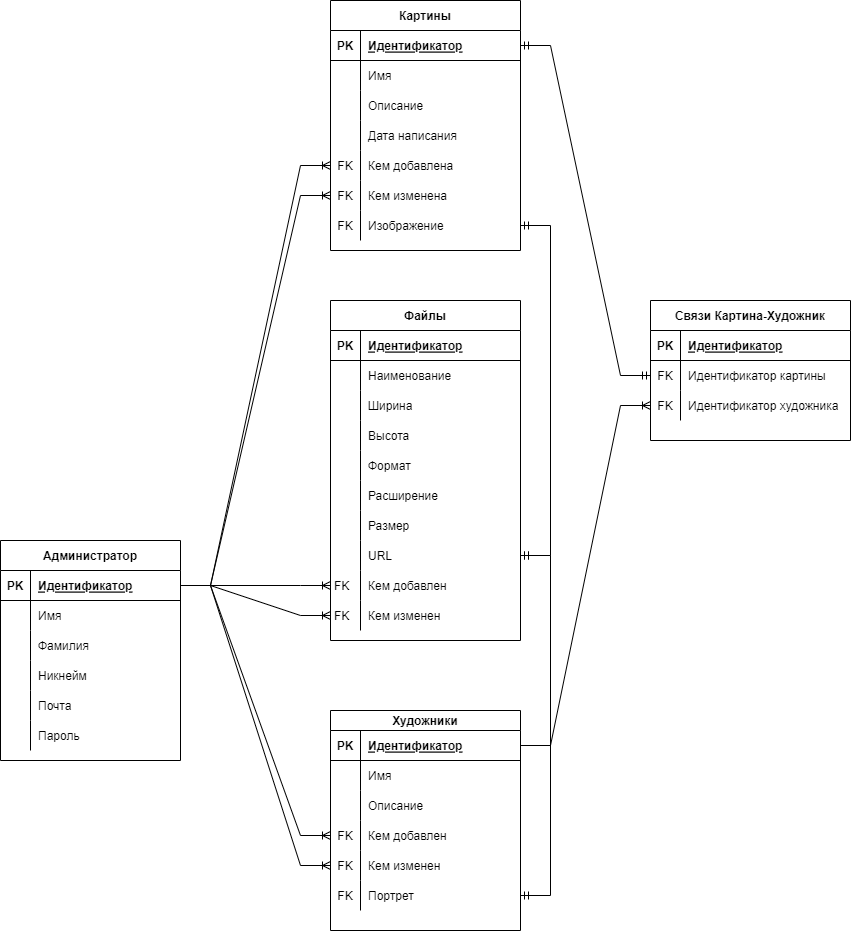


Рисунок 16. Даталогическая модель БД.

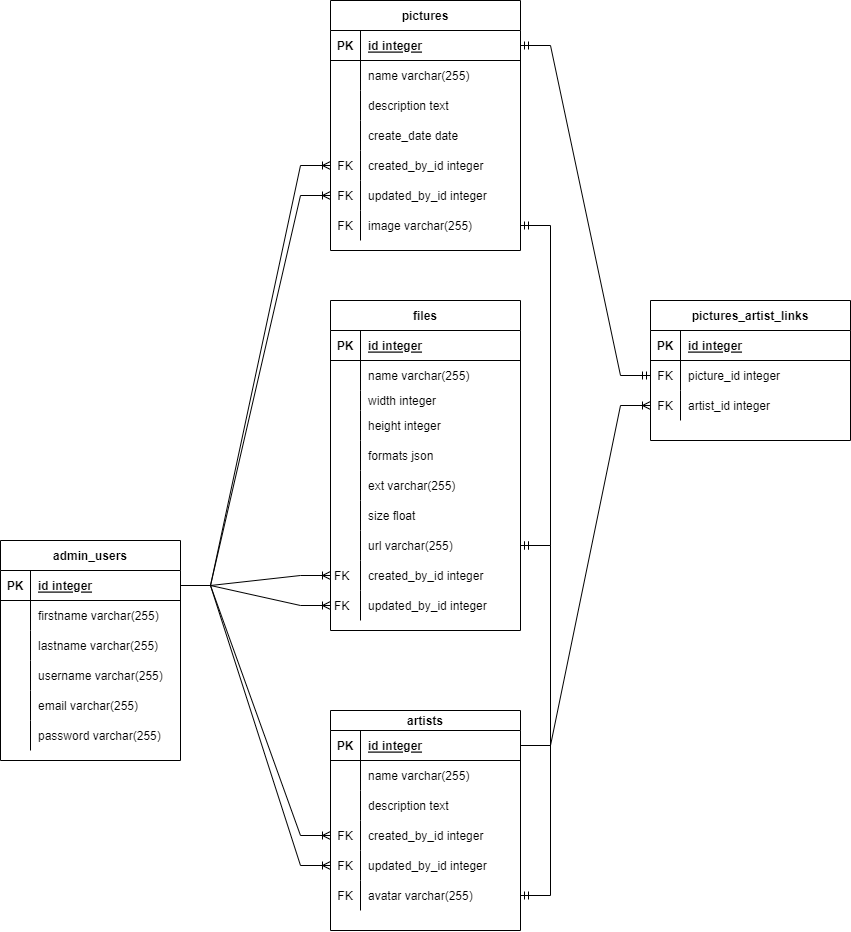


Рисунок 17. Физическая модель БД.

## Выводы

В данной главе было произведено исследование предметной области на предмет автоматизации процессов, а также, рассмотрены концептуальные модели системы, необходимые для ее разработки.

1. Реализация проекта

## Реализация front-end части информационной системы

Для реализация пользовательского интерфейса была использована JavaScript библиотека React. Разработаны компоненты и страницы.

Далее перечислены компоненты пользовательского интерфейса, листинг кода которых доступен в соответствующих приложениях:

1. Header – приложение 1. Пункт 1;
2. Picture – приложение 1. Пункт 2;
3. SearchBar – приложение 1. Пункт 3;
4. App – приложение 1. Пункт 4.

Далее перечисляются страницы, использующие вышеперечисленные компоненты пользовательского интерфейса, листинг кода которых так же доступен в соответствующих приложениях:

1. Home - приложение 2. Пункт 1;
2. PictureDetail - приложение 2. Пункт 2;
3. Search – приложение 2. Пункт 3.

Точкой входа для выполнения веб-приложения является файл index.js. Листинг кода представлен в разделе «Приложение 3».

## Реализация back-end части информационной системы

Для реализации использовалась система управления контентом (headless-CMS) под названием Strapi, в которой создается API приложения. В качестве архитектурного стиля API был выбран REST.

Для url-адреса веб-приложения определены конечные точки (endpoint), к которым можно обращаться с помощью различных методов сетевых запросов. Данные запросы обрабатываются посредством методов контроллера конкретной модели объекта.

Для каждого контроллера модели определены следующие методы:

1. Find – получение всех объектов данной конечной точки:

async find(ctx) {

const { query } = ctx;

const { results, pagination } = await strapi.service(uid).find(query);

const sanitizedResults = await this.sanitizeOutput(results, ctx);

return this.transformResponse(sanitizedResults, { pagination });

}

1. FindOne – получение одного объекта данной конечной точки:

async findOne(ctx) {

const { id } = ctx.params;

const { query } = ctx;

const entity = await strapi.service(uid).findOne(id, query);

const sanitizedEntity = await this.sanitizeOutput(entity, ctx);

return this.transformResponse(sanitizedEntity);

}

1. Create – создание нового объекта:

async create(ctx) {

const { query } = ctx.request;

const { data, files } = parseBody(ctx);

if (!isObject(data)) {

throw new ValidationError('Missing "data" payload in the request body');

}

const sanitizedInputData = await this.sanitizeInput(data, ctx);

const entity = await strapi

.service(uid)

.create({ ...query, data: sanitizedInputData, files });

const sanitizedEntity = await this.sanitizeOutput(entity, ctx);

return this.transformResponse(sanitizedEntity);

}

1. Update –обновление объекта.

async update(ctx) {

const { id } = ctx.params;

const { query } = ctx.request;

const { data, files } = parseBody(ctx);

if (!isObject(data)) {

throw new ValidationError('Missing "data" payload in the request body');

}

const sanitizedInputData = await this.sanitizeInput(data, ctx);

const entity = await strapi

.service(uid)

.update(id, { ...query, data: sanitizedInputData, files });

const sanitizedEntity = await this.sanitizeOutput(entity, ctx);

return this.transformResponse(sanitizedEntity);

}

1. Delete –удаление объекта:

async delete(ctx) {

const { id } = ctx.params;

const { query } = ctx;

const entity = await strapi.service(uid).delete(id, query);

const sanitizedEntity = await this.sanitizeOutput(entity, ctx);

return this.transformResponse(sanitizedEntity);

}

## Реализация базы данных

Для хранения данных в системе используется встраиваемая СУБД SQLite.

Подключение к БД происходит через конфигурационный файл database.js.

Листинг кода database.js:

module.exports = ({ env }) => ({

connection: {

client: 'sqlite',

connection: {

filename: path.join(\_\_dirname, '..', env('DATABASE\_FILENAME', '.tmp/data.db')),

},

useNullAsDefault: true,

},

});

Запросы к БД создаются с помощью генератора SQL запросов Knex.js.

Заключение

В процессе данной курсовой работы была выполнена задача по созданию проекта «Цифровая галерея». После выполнений всех этапов разработки можно заключить, что информационная система готова к эксплуатации.

### СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ. И ЛИТЕРАТУРЫ

1. Strapi v4 developer documentation [Электронный ресурс]. – Режим доступа: <https://docs.strapi.io/developer-docs/latest/getting-started/introduction.html>, свободный. – (Дата обращения: 15.12.2022).
2. React. JavaScript-библиотека для создания пользовательских интерфейсов [Электронный ресурс]. – Режим доступа: <https://ru.reactjs.org/docs/getting-started.html> , свободный. – (Дата обращения: 16.12.2022).
3. Бибо Бер , Кац Иегуда jQuery. Подробное руководство по продвинутому JavaScript; Символ-плюс - М., 2017. - 624 c.
4. Дронов Владимир JavaScript и AJAX в Web-дизайне; БХВ-Петербург - М., 2015. - 736 c.
5. Дронов Владимир JavaScript. Народные советы; БХВ-Петербург - М., 2016. - 458 c.
6. Дунаев Вадим JavaScript. Самоучитель; Питер - М., 2015. - 400 c.
7. Дунаев Вадим HTML, скрипты и стили; БХВ-Петербург - М., 2015. - 816 c.
8. Изучаем Node.js; Питер - М., 2015. - 400 c.
9. Клименко Роман Веб-мастеринг на 100%; Питер - М., 2015. - 920 c.
10. Климов Александр JavaScript на примерах; БХВ-Петербург - М., 2017. - 812 c.
11. Крокфорд Д. JavaScript. Сильные стороны; Питер - М., 2016. - 262 c.
12. Лазаро Исси Коэн, Джозеф Исси Коэн Полный справочник по HTML, CSS и JavaScript; ЭКОМ Паблишерз - М., 2016. - 311 c.
13. Макфарланд Дэвид JavaScript. Подробное руководство; Эксмо - М., 2015. - 608 c.
14. Никсон Робин Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5; Питер - М., 2016. - 768 c.
15. Прохоренок Н. А. Python. Самое необходимое; БХВ-Петербург - М., 2015. - 416 c.
16. Резиг Джон , Бибо Беэр Секреты JavaScript ниндзя; Вильямс - М., 2015. - 416 c.
17. Роббинс Дженнифер HTML5, CSS3 и JavaScript. Исчерпывающее руководство (+ DVD-ROM); Эксмо - М., 2017. - 528 c.
18. Фримен Адам jQuery для профессионалов; Вильямс - М., 2015. - 960 c.
19. Херман Дэвид Сила JavaScript. 68 способов эффективного использования JS; Питер - М., 2015. - 952 c.
20. Чаффер Д. Изучаем jQuery 1.3. Эффективная веб-разработка на JavaScript; Символ-плюс - М., 2015. - 391 c.
21. Knex Query Builder [Электронный ресурс]. – Режим доступа: <https://knexjs.org/guide/query-builder.html>, свободный. – (Дата обращения: 12.01.2023).
22. Andy, Harris HTML, XHTML and CSS All–In–One For Dummies® / Andy Harris. - Москва: Наука, 2014. - 173 c.
23. Ben, Henick HTML & CSS – The Good Parts / Ben Henick. - Москва: СИНТЕГ, 2013. - 350 c.
24. Ed, Tittel HTML, XHTML & CSS For Dummies® / Ed Tittel. - Москва: Гостехиздат, 2012. - 416 c.
25. Гаевский, А.Ю. 100% самоучитель. Создание Web-страниц и Web-сайтов. HTML и JavaScript / А.Ю. Гаевский, В.А. Романовский. - М.: Триумф, 2014. - 464 c.
26. Общие сведения о веб-приложениях [Электронный ресурс]. – Режим доступа: <https://helpx.adobe.com/ru/dreamweaver/using/web-applications.html>, свободный. – (Дата обращения: 25.01.2023).
27. Что такое веб-приложение простыми словами: виды и алгоритм разработки [Электронный ресурс]. – Режим доступа: <https://vc.ru/dev/397220-chto-takoe-veb-prilozhenie-prostymi-slovami-vidy-i-algoritm-razrabotki>, свободный. – (Дата обращения: 26.01.2023).

Приложение 1

1. …Components/Header.js:

import './header.css'

import SearchBar from '../components/SearchBar'

import {useNavigate} from 'react-router-dom'

function Header({searchValue, setSearchValue}) {

  const navigate = useNavigate()

  return (

    <header>

      <div className='wrapper'>

        <div className='logo' onClick={() => navigate('/')}>

          <h2>Цифровая галерея</h2>

        </div>

        <SearchBar searchValue={searchValue} setSearchValue={setSearchValue}/>

      </div>

    </header>

  )

}

export default Header

1. …Components/Picture.js:

import './picture.css'

import { useNavigate } from 'react-router-dom'

function Picture({title, image, id, artist, onClick = () => {}}) {

  const navigate = useNavigate();

  return (

    <div className='picture' onClick={() => {navigate(`/detail/${id}`, {

      state: {id}

    }); onClick()}}>

      <img src={image} alt='Картина'/>

      <div className='picture\_desk'>

        <h4>{title}</h4>

        <div className='picture\_artist'>

          <span>{artist}</span>

          {/\* By: {authors?.map((author, i) => (

            <span key={i}>{author}</span>

          ))} \*/}

        </div>

      </div>

    </div>

  )

}

export default Picture;

1. …Components/SearchBar.js:

import './searchBar.css'

import {SearchRounded } from '@mui/icons-material';

import { Button } from '@mui/material';

import { useNavigate } from 'react-router-dom';

import { useEffect } from 'react';

function SearchBar({searchValue, setSearchValue}) {

  const navigate = useNavigate()

  const handleSubmit = (e) => {

    e.preventDefault()

    if(searchValue !== ''){

      navigate(`/search/${searchValue}`)

    }

  }

  useEffect(() => {

    setSearchValue(localStorage.getItem("searchValue"))

  }, [])

  return (

    <div  className='searchContainer'>

      <form className='search' onSubmit={handleSubmit}>

        <input

          type='text'

          placeholder='Найти картину'

          value={searchValue}

          onChange={e => {

            setSearchValue(e.target.value)

            localStorage.setItem("searchValue", e.target.value);

          }}/>

        {searchValue?.length >= 1 &&

          <span

            className='search\_\_cancel'

            onClick={() => setSearchValue('')}>x</span>

        }

        <Button

          id='search\_\_iconContainer'

          variant='contained'

          disableElevation

          onClick={handleSubmit}

        >

        <SearchRounded className='search\_\_icon'/>

        </Button>

      </form>

    </div>

  )

}

export default SearchBar

1. App.js

import './App.css';

import { Routes, Route } from "react-router-dom"

import Home from './pages/Home';

import PictureDetail from './pages/PictureDetail';

import Header from './components/Header';

import Search from './pages/Search';

import { useState } from 'react';

function App() {

  const [searchValue, setSearchValue] = useState('')

  return (

    <div className="App">

      <Header searchValue={searchValue} setSearchValue={setSearchValue}/>

      <Routes>

        <Route path='/' exact element={<Home searchValue = {searchValue} setSearchValue={setSearchValue}/>}/>

        <Route path='/detail/:id' element={<PictureDetail />}/>

        <Route path='/search/:query' element={<Search searchValue={searchValue}/>}/>

      </Routes>

    </div>

  );

}

export default App;

Приложение 2

1. Pages/Home.js

import { useEffect, useState } from 'react'

import Picture from '../components/Picture'

import './home.css'

//import {AllPictures} from '../data.js'

import axios from 'axios'

function Home({searchValue, setSearchValue}) {

  useEffect(() => {

    setSearchValue('');

  }, [])

  const [pictures, setPictures] = useState([]);

  const URL = "http://localhost:1337/api/pictures?populate=\*";

  const imageURL = "http://localhost:1337"

  const fetchData = async () => {

    let data;

    const result = await axios.get(URL);

    data = result.data.data;

    setPictures(data);

  }

  useEffect(() => {

    fetchData();

  });

  //console.log(JSON.stringify(pictures[0]?.attributes.image.data[0].attributes.formats.small.url));

  return (

    <div className='home'>

      <div className='hero wrapper'>

        <div className='wrapper'>

          <div className='hero\_\_text'>

            <h2>Добро пожаловать</h2>

            <p>Здесь представлены произведения искусства Бирского филиала УУНиТ</p>

          </div>

          <img src='/picHero2.jpg' alt='hero' />

        </div>

      </div>

      <div className='allPictures wrapper'>

        <h2>Все картины</h2>

        <div className='pictures'>

          {pictures?.map((pic) => (

            <Picture

              key={pic.id}

              title={pic.attributes.name}

              image={imageURL+pic.attributes.image.data[0].attributes.formats.small.url}

              id={pic.id}

              artist = {pic?.attributes?.artist?.data?.attributes.name}

            />

          ))}

        </div>

      </div>

    </div>

  )

}

export default Home

1. Pages/PictureDetail.js

import './pictureDetail.css'

import {useLocation} from 'react-router-dom'

import { useEffect, useState } from 'react'

import axios from 'axios'

function PictureDetail() {

  const [picture, setPicture] = useState({});

  const {state} = useLocation();

  //console.log('id = '+ state.id);

  const URL = `http://localhost:1337/api/pictures/${state.id}?populate=\*`;

  const imageURL = "http://localhost:1337";

  // useEffect(() => {

  //   const bookData = Allbooks.filter((book) => (

  //     book.id === params.id

  //   ))

  //   setBook(bookData[0])

  // }, [params])

  const fetchData = async () => {

    let data;

    const result = await axios.get(URL).catch(

      (error) => {

        if (error.response) {

          // The request was made and the server responded with a status code

          // that falls out of the range of 2xx

          console.log(error.response.data);

          console.log(error.response.status);

          console.log(error.response.headers);

        } else if (error.request) {

          // The request was made but no response was received

          // `error.request` is an instance of XMLHttpRequest in the browser and an instance of

          // http.ClientRequest in node.js

          console.log(error.request);

        } else {

          // Something happened in setting up the request that triggered an Error

          console.log('Error', error.message);

        }

        console.log(error.config);

      }

    );

    data = result.data.data;

    setPicture(data);

  }

  useEffect(() => {

    fetchData();

  });

  console.log(JSON.stringify(picture));

  //проверка на наличие изображения высокого/среднего/низкого качества

  let imageFormats = picture.attributes?.image.data[0].attributes.formats;

  let image = '';

  if (imageFormats?.large) {

    image = imageURL+imageFormats.large.url;

  }

  else if (imageFormats?.medium) {

    image = imageURL+imageFormats.medium.url;

  }

  else image = imageURL+imageFormats?.small.url;

  console.log(image);

  //--------------------------------------------------

  return (

    <div className='pictureDetail wrapper'>

      <div className='pictureDetail\_\_topDec'>

        <img src={image} alt='Картина'/>

        <div className='pictureDetail\_\_textBlock1'>

          <h3>{picture.attributes?.name}</h3>

          <div className='pictureDetail\_\_artist'>

            {picture.attributes?.artist.data.attributes.name}

          </div>

        </div>

      </div>

      <div className='pictureDetail\_\_bottomDec'>

        <h4>О картине</h4>

        <p>{picture.attributes?.description}</p>

      </div>

    </div>

  )

}

export default PictureDetail;

1. Pages/Search.js

import './search.css'

import MeiliSearch from 'meilisearch'

import {useEffect, useState} from 'react'

import Picture from '../components/Picture'

function Search({searchValue}) {

  const [pictures, setPictures] = useState([])

  useEffect(() => {

      if (searchValue !== '') {

      const fetchData = async () => {

        const client = new MeiliSearch({

          host: 'http://127.0.0.1:7700',

        })

        const index = await client.getIndex('picture')

        const picturesData = await index.search(searchValue)

        setPictures(picturesData.hits)

      }

      fetchData()

    }

  }, [searchValue])

  console.log(JSON.stringify(pictures));

  const imageURL = "http://localhost:1337";

  return (

    <div className='searchPage wrapper'>

      <div className='searchPage\_\_resultInfo'>

        <p>Результаты поиска:</p>

        {/\* <h4>{searchValue}</h4> \*/}

      </div>

      <div className='pictures'>

        {pictures?.map((pic) => (

          <Picture

            key={pic.id}

            title={pic.name}

            image={imageURL+pic.image[0].formats.medium?.url}

            id={pic.id}

            artist = {pic.artist?.name}

          />

        ))}

      </div>

    </div>

  )

}

export default Search;

Приложение 3

/Index.js

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import { BrowserRouter } from "react-router-dom";

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

  <BrowserRouter>

    <App />

  </BrowserRouter>

);