

Аннотация

Тема выпускной квалификационной работы «Разработка информационной системы для музея МБУ «Школа №4 имени Н.В. Абрамова»».

Ключевые слова: информационная система для музея, логическое моделирование информационной системы, база данных PostgreSQL, Node.js, React, ReactAdmin.

Данная выпускная квалификационная работа посвящена разработке информационной системы для школьного музея.

Работа состоит из трех глав. В первой главе проведен анализ работ школьного музея с предоставлением декомпозиции бизнес-процессов и разработана модель «КАК ДОЛЖНО БЫТЬ». Во второй главе описан этап логического проектирования разрабатываемой информационной системы с использованием объектно-ориентированного подхода. В третьей главе рассмотрена разработка информационной системы, а именно ее компонентов – сервера, панели администратора, базы данных. Программный код информационной системы изоморфный, то есть реализация всей информационной системы производилась на одном языке программирования. Также в третьей главе проводится описание процесса тестирования и представление результатов тестирования. В заключении показаны выводы и результаты о проделанной работе.

Результатом выпускной квалификационной работы будет являться информационная система для организации поисковой деятельности по сбору дополнительного материала об истории образовательных учреждений, вовлечение обучающихся в социально-значимую деятельность.

В выпускной квалификационной работе содержится 59 страниц с приложениями, 15 рисунков, 12 таблиц, 21 литературный источник.

Abstract

The title of the graduation work is Development of an information system for the school museum of municipal budgetary institution school No. 4 named after N. V. Abramov.

The Museum of Education of school No. 4 consists of several sections, one of which is a virtual database. The virtual database is an information system with data on the history of education, information about educational institutions and information about the exhibits of the school museum.

The aim of the work is developing an information system for the museum.

The object of the graduation work is the museum activities.

The subject of the graduation work is the automation of the processes of storing, processing and using information.

The work touches upon the analysis of an existing information system using BPMN notation, modeling of a new information system using the UML methodology, the development process and capabilities of a new information system, manual testing, implementation of the information system in use.

As a result, all the tasks were completed, and the created information system was put into use in the museum. After increasing the page loading speed and adapting the information system for mobile devices, you can use it not only in the museum classroom on the interactive panel, but also from your home computer and even your phone. The information system administrator now has more options for editing information.

The developed information system has a good extensibility, so it will not take much time to refine it to meet new needs.

Оглавление

Введение.....	4
Глава 1 Функциональное моделирование информационной системы для музея	6
1.1 Анализ деятельности научно-просветительского комплекса «Музей образования».....	6
1.2 Концептуальное моделирование информационной системы для музея	8
1.3 Постановка задачи на разработку проекта информационной системы для музея.....	14
Глава 2 Логическое проектирование информационной системы для музея	17
2.1 Выбор технологии логического моделирования информационной системы для музея.....	17
2.2 Разработка логической модели информационной системы для музея и ее описание	18
2.3 Проектирование базы данных информационной системы для музея	25
Глава 3 Физическое проектирование информационной системы для музея	28
3.1 Выбор архитектуры для информационной системы для музея	28
3.2 Выбор технологии разработки программного обеспечения для информационной системы для музея.....	29
3.3 Выбор системы управления базой данных для информационной системы для музея.....	32
3.4 Разработка физической модели данных для информационной системы для музея.....	33
3.5 Разработка программного обеспечения информационной системы для музея	34
3.6 Разработка диаграммы развертывания информационной системы.	42
3.7 Тестирование информационной системы.....	43

Заключение	47
Список используемой литературы	48
Приложение А Модель «КАК ЕСТЬ» процесса работы с панелью администратора	51
Приложение Б Модель «КАК ЕСТЬ» процесса использования информационной системы в образовательных целях	52
Приложение В Модель «КАК ДОЛЖНО БЫТЬ» процесса работы с панелью администратора	53
Приложение Г Модель «КАК ДОЛЖНО БЫТЬ» процесса использования информационной системы в образовательных целях	54
Приложение Д Код создания сервера	55
Приложение Е Код из файла конфигурации package.json	57
Приложение Ж Тест-кейсы	59

Введение

Информационные технологии прочно входят в музейную жизнь. Они выступают в качестве эффективного инструмента музейного развития, позволяя музеям по-новому позиционировать себя в культурном поле, вести более активную культурную политику.

Централизованное хранение и использование информации является актуальной задачей на сегодня. Благодаря централизованному хранению информации пользователь сможет получить к ней доступ с любого устройства – смартфона, планшета, компьютера, при этом на всех устройствах будет идентичная информация.

Проектирование и реализация информационной системы (ИС) выполняется по заказу МБУ «Школа №4 имени Н.В. Абрамова». Заказчику требуется ИС для хранения музейной информации и использования этой информации в образовательных и экскурсионных целях. Для решения поставленной задачи предложено создать веб-приложение и мобильное приложение.

Объектом исследования данной работы является деятельность музея в МБУ «Школа №4 имени Н.В. Абрамова».

Предметом исследования является автоматизация процессов хранения, обработки и использования информации.

Целью данной работы является разработка информационной системы для музея МБУ «Школа №4 имени Н.В. Абрамова». Практическая значимость разработанной ИС заключается в ее использовании в образовательных целях и целях сбора и хранения информации.

Для достижения поставленной цели требуется решение следующих задач:

- выделить основные бизнес-процессы в школьном музее и провести их анализ,

- сформулировать постановку задачи на разработку ИС с описанием требований,
- провести проектирование информационной системы,
- выбрать средства реализации информационной системы,
- реализовать информационную систему,
- провести тестирование ИС,
- внедрить информационную систему в использование.

В первой главе описана исследуемая предметная область, выбрана технология концептуального моделирования, разработана модель «КАК ЕСТЬ», выявлены процессы, подлежащие автоматизации, поставлена задача на разработку новой ИС и разработана модель бизнес-процесса «КАК ДОЛЖНО БЫТЬ».

Во второй главе выбраны технологии для логического моделирования, составлена логическая модель ИС, спроектирована логическая модель базы данных и определены требования к аппаратно-программному обеспечению ИС.

В третьей главе рассмотрен этап физического проектирования ИС. Выбрана архитектура, технологии разработки программного обеспечения, система управления базами данных (СУБД). На основе логической модели данных разработана физическая модель данных, описана разработка базы данных, сервера и клиентской части ИС. Для внедрения информационной системы разработана диаграмма развертывания. Проведено ручное тестирование ИС.

Глава 1 Функциональное моделирование информационной системы для музея

1.1 Анализ деятельности научно-просветительского комплекса «Музей образования»

Исследуемая предметная область – виртуальная база данных научно-просветительского комплекса «Музей образования» далее именуемая музеем. Музей организован на базе МБУ «Школа №4 имени Н.В. Абрамова» и является ее подразделением (рисунок 1).

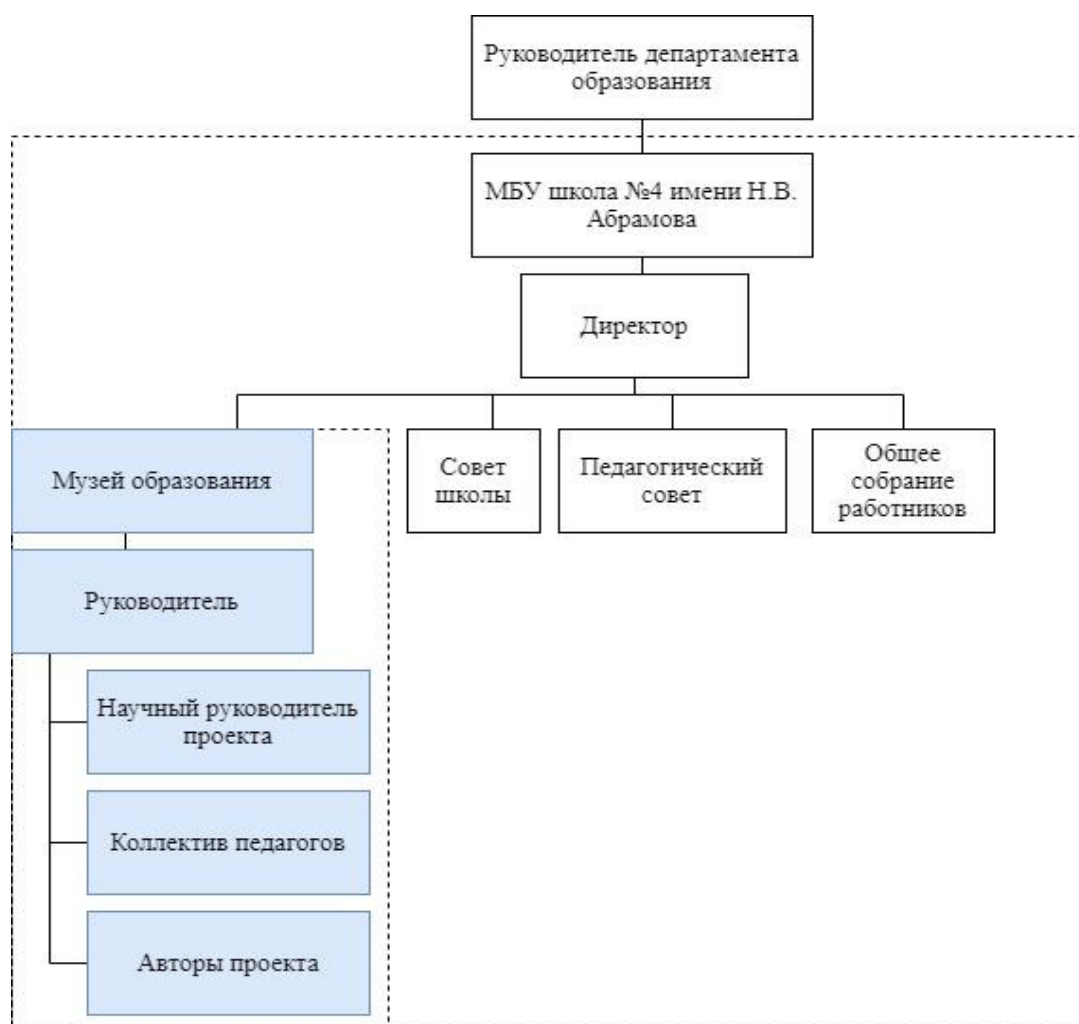


Рисунок 1 – Организационная структура музея образования

Учреждение находится в ведомственном подчинении Департамента образования администрации городского округа Тольятти [12]. В кадровое обеспечение музея входят: руководитель школьного музея, коллектив педагогов, научный руководитель проекта, авторы проекта.

Цель функционирования музея – создание современного многофункционального интерактивного пространства как центра исследовательской и проектной деятельности учащихся и преподавателей, способствующего популяризации науки [8].

Задачи музея:

- организация поисковой деятельности по сбору дополнительного материала об истории образовательных учреждений, вовлечение обучающихся в социально-значимую деятельность;
- организация научной деятельности по разработке педагогами и учащимися научных тем по биологии, нейробиологии, физике, химии в рамках создания научного пространства «Лаборатория знаний»;
- расширение компетенции педагогов, овладение навыками проектной технологии и деятельностного подхода, музейной педагогики;
- развитие у обучающихся гражданско-патриотических качеств гражданина России, чувства уважения к старшему поколению, посредством участия в организации Клуба ветеранов образования;
- создание скульптуры «Первой учительницы».

Музей образования стал центром дополнительного образования, гражданского и патриотического воспитания, местом изучения истории образования Ставрополя-Тольятти, а также центром становления личности учащегося, ее развитием и совершенствованием. Идея создания Музея образования значительно расширяет рамки традиционного школьного музея, формирует круг своих постоянных посетителей – как школьников, так и взрослых горожан, способствует развитию информационной культуры и

максимальному включению в совместную проектную деятельность. «Музей образования» состоит из следующих разделов:

- виртуальная база данных «История образования Ставрополя-Тольятти»,
- скульптурная композиция «Первая учительница»,
- интерактивное пространство «Лаборатория знаний»,
- клуб ветеранов образования.

Виртуальная база данных «История образования Ставрополя-Тольятти» – это информационная система с информацией об истории образования, образовательных учреждениях и об экспонатах музея образования. Она должна иметь понятный пользователю интерфейс, так как ей преимущественно будут пользоваться школьники и педагоги, а также содержать функционал для ее пополнения информацией.

1.2 Концептуальное моделирование информационной системы для музея

1.2.1 Выбор технологии концептуального моделирования информационной системы для музея

В настоящее время существует несколько популярных методологий и технологий для концептуального моделирования, такие как SADT, BPMN, UML.

SADT – это методология, разработанная специально для того, чтобы облегчить описание и понимание искусственных систем, попадающих в разряд средней сложности [4]. С точки зрения SADT модель, построенная в этой методологии, может быть сосредоточена либо на функциях системы, либо на ее объектах.

BPMN – нотация, позволяющая описать внутренние бизнес-процессы предприятия и стандартизировать их описание [11].

UML – помогает определять, визуализировать и документировать модели программных систем, включая их структуру и дизайн, таким образом, чтобы они отвечали всем этим требованиям [17].

Проведем сравнение каждой методологии по критериям:

- возможность отражения функциональной и поведенческой составляющей предметной области,
- выразительность модели,
- оптимальное количество типов моделей (диаграмм).

В таблице 1 описана каждая методология и нотация по заданным критериям.

Таблица 1 – Сравнение методологий и нотаций для проектирования

Критерии	SADT	BPMN	UML
Возможность отражения функциональной и поведенческой составляющей предметной области	Содержит статические и причинно-следственные связи и события, например, описание структуры, потоки информации [2].	Описывает бизнес-процессы в блок-схеме	Описывает статическое и динамическое состояние системы.
Выразительность модели	На диаграмме присутствуют объекты, требующие предварительного изучения.	Позволяет создавать сложные и простые блок-схемы	Много моделей, описывающих систему с разных сторон.
Оптимальное количество типов моделей (диаграмм)	Поддерживает диаграммы IDEF0, DFD, IDEF3.	Блок-схема из пяти основных категорий элементов	Поддерживает 12 типов диаграмм.

Методологии UML имеют возможность отобразить, как пользователь взаимодействует с ИС, каким образом обрабатываются данные и как сохраняются в базу данных, из каких сущностей состоит ИС. Однако методология UML не позволяет отразить в одной модели процессы, подлежащие автоматизации. Методология SADT подходит для анализа процессов, но не подходит для логического моделирования, так как в ней

нельзя в полной мере описать ИС. Так же SADT изначально не рассчитывалась на моделирование процессов и информационных систем. Нотация BPMN имеет достаточно простой набор объектов для отображения бизнес-процессов, что позволяет просто выявлять проблемы в процессах. Нотация BPMN показывает потенциальные области процесса для улучшения.

Поэтому для анализа бизнес процессов использована нотация BPMN, в ней построены модели «КАК ЕСТЬ» и «КАК ДОЛЖНО БЫТЬ».

1.2.2 Моделирование бизнес-процессов информационной системы музея для постановки задачи автоматизированного варианта решения

При разработке ИС выделяют несколько этапов жизненного цикла:

- формирование требований,
- проектирование,
- реализация,
- внедрение,
- эксплуатация.

Формирование требований включает в себя планирование работ, проведение обследования деятельности автоматизированного объекта, построение моделей деятельности организации на основе результатов обследования [5]. Для понимания процессов автоматизации необходимо построить модели «КАК ЕСТЬ» и «КАК ДОЛЖНО БЫТЬ».

Модель «КАК ЕСТЬ» отражает ИС на момент обследования организации. Модель «КАК ДОЛЖНО БЫТЬ» отображает, как должна выглядеть архитектура ИС музея после внесения изменений в существующую ИС.

Для построения моделей существует достаточно много программных решений. Так как в качестве методологии для проектирования выбрана методология UML, а для анализа нотация BPMN, то рассмотрим для них наиболее популярные программные решения – StarUML, Visio, Draw.io.

Microsoft Visio – решение для построения диаграмм от Microsoft. Векторный графический редактор, редактор диаграмм и блок-схем для Windows [19].

StarUML – программный инструмент моделирования, который поддерживает UML. StarUML ориентирован на UML версии 1.4 и поддерживает одиннадцать различных типов диаграмм, принятых в нотации UML 2.0 [14]. Он активно поддерживает подход MDA, реализуя концепцию профилей UML.

Draw.io – бесплатное приложение, предназначенное для моделирования диаграмм и блок-схем бизнес-процессов.

Сравнение программных решений было проведено по следующим критериям:

- создание программного кода на основе диаграмм,
- поддержка операционными системами,
- создание диаграмм на основе программного кода,
- набор фигур для методологий.

Результаты сравнения программных средств для реализации диаграмм приведены в таблице (таблица 2).

Таблица 2 – Сравнение программных средств

Критерии	Microsoft Visio	StarUML	Draw.io
Создание программного кода на основе диаграмм	Нет	Есть	Нет
Поддержка операционными системами	Microsoft Windows	Cross-platform	Cross-platform
Создание диаграмм на основе программного кода	Нет	Нет	Нет
Набор фигур для методологий	SADT, UML	UML	UML, SADT, BPMN

Анализ бизнес процессов производился в программном решении Draw.io, оно содержит все необходимые объекты для рисования блок-схем из нотации BPMN.

1.2.3 Разработка и анализ модели бизнес-процесса «КАК ЕСТЬ»

Модель «КАК ЕСТЬ» построена, чтобы систематизировать протекающие процессы в ИС и выявить процессы, подлежащие автоматизации. В качестве диаграммы «КАК ЕСТЬ» выбрана BPMN модель. Модель составлена для двух вариантов использования, так как информационная система может быть использована для нескольких независимых процессов. На рисунке А.1 представлен процесс работы администратора в панели администратора. Из диаграммы видно два главных минуса существующей ИС:

- если пользователь не прошел аутентификацию, то высветится белый экран без понятного вывода ошибок;
- администратор ИС может редактировать только две сущности.

Процесс работы с информационной системой школьников, преподавателей и посетителей музея показан на рисунке Б.1. Из диаграммы видно, что все роли имеют возможность воспользоваться ИС только в рабочее время музея.

1.2.4 Обоснование необходимости автоматизированного варианта решения и формирование требований к информационной системе для музея

Большая часть информации на сайте ИС является статичной, так как создать новую страницу можно только для учебного заведения и ветерана образования, происходит это из-за того, что в панели администратора отсутствует функция редактирования информации. Так как научно-просветительский комплекс постоянно развивается, то на сайте требуется добавлять новые разделы, что сейчас невозможно без вмешательства разработчика информационной системы.

Для разрабатываемой ИС составлены требования, учитывающие все недостатки существующей ИС.

Требования к ИС в классификации FURPS+:

а) functionality:

- 1) добавление любых страниц;
- 2) создания страниц с любой комбинацией блоков;
- 3) просмотр страниц сайта;
- 4) управление ключевой информацией разработчиком из панели администратора;
- 5) редактирование сущностей в панели администратора;
- 6) генерация qr-кода для поиска экспоната музея;

б) usability:

- 1) доступный интерфейс в панели администратора;

в) reliability:

- 1) автоматический запуск системы после сбоя в работе;
- 2) проверка перед запуском, что все ключевые записи в базе данных созданы;
- 3) создание ключевых записей в базе данных;
- 4) логирование действий в панели администратора;
- 5) несколько ролей в панели администратора;

г) performance:

- 1) добавление разделов, страниц без вмешательства разработчика;

д) supportability:

- 1) поддержка панели администратора браузерами:
 - Google Chrome, минимальная версия 21.

Требования выявлены после анализа недостатков существующей ИС и исходя из пожеланий заказчика.

1.3 Постановка задачи на разработку проекта информационной системы для музея

Назначением реализации работы по созданию ИС для музея служит:

- а) увеличение функциональных возможностей панели администратора:
 - разделение ролей;
 - просмотр файлов логирования;
 - редактирование всех сущностей базы данных;
 - создание новых страниц и разделов на сайте;
 - генерирование qr-кодов для экспонатов музея и сопоставление им записи об экспонате в базе данных;
 - возможность разработчику редактировать ключевую информацию из панели администратора.
- б) адаптация панели администратора под мобильные устройства, планшеты, компьютеры;
- в) проверка и создание сервером ключевых записей в базе данных.

Перечисленные выше требования добавляют новые функциональные возможности к уже существующей ИС.

Изменения в функциональных подразделениях, связанных со сбором, обработкой и выдачей информации включают добавление разделения ролей в панели администратора. В разрабатываемой ИС появилась роль «Разработчик», который может вносить изменения в ключевые записи в базе данных. Разработчик может создавать новых пользователей, задавать им пароли для доступа к панели администратора. Важной функцией разработчика является доступ к файлам логирования и выявление ошибок в работе ИС, а также проверка действий пользователей в панели администратора.

После выявления требований к ИС, можно разделить ее на несколько функциональных подсистем:

- панель администратора,
- сервер,
- сайт,
- мобильное приложение.

В выпускной квалификационной работе будут рассмотрены компоненты «панель администратора» и «сервер».

Этапы решения выявленных задач:

- проектирование новой ИС;
- выбор технологий для реализации ИС;
- реализация каждой подсистемы, входящей в ИС виртуальной базы данных музея;
- тестирование ИС.

На проведение всех этапов реализации ИС выделяется 6 месяцев, график реализации ИС, представлен на диаграмме Ганта (рисунок 2).

	сентябрь, 2020				октябрь, 2020				ноябрь, 2020				декабрь, 2020				январь, 2021				февраль, 2021				март, 2021				апрель, 2021	
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	
Анализ существующей ИС	■	■																												
Выявление требований и утверждение их с заказчиком			■	■	■	■	■	■																						
Проектирование новой ИС							■	■	■	■	■	■																		
Создание прототипа ИС											■	■																		
Создание цветного макета													■	■																
Выбор технологий для реализации ИС													■	■																
Реализация каждой подсистемы, входящей в ИС музея																	■	■	■	■	■	■	■	■	■	■	■	■		
Тестирование ИС																													■	■

Рисунок 2– Сроки реализации информационной системы

В результате реализации ИС можно запустить на выделенном виртуальном сервере. Все компоненты ИС взаимодействуют через подсистему «Сервер».

При разработке ИС используются свободно распространяемые средства разработки, такие как база данных PostgreSQL, библиотека React, фреймворки Angular, Vue, платформа Node.js.

В результате произведенного анализа текущего состояния ИС, выявлены новые требования и процессы, подлежащие автоматизации. Для того чтобы визуально представить, как должны протекать процессы при работе в панели администратора, составлена BPMN диаграмма «КАК ДОЛЖНО БЫТЬ» (рисунок В.1). Зеленым цветом на диаграмме показаны новые части процесса работы в панели администратора. При неудачной попытке аутентификации пользователь понимает, что произошло, благодаря понятному для него сообщению. Создавать, редактировать и удалять можно любую сущность из базы данных, если эти действия не ограничены правами пользователя.

Модернизированный процесс использования ИС показан на рисунке Г.1. Из диаграммы видно, что у пользователей ИС появилось несколько вариантов получить информацию – теперь это можно сделать удаленно, а не только в классе музея.

Выводы по главе 1

В данной главе была исследована предметная область – виртуальная база данных музея.

С учетом созданных концептуальных моделей и проведенного анализа были сформулированы основные требования к разрабатываемому ПО, цель разработки системы, а также определены необходимые функциональные возможности и особенности работы.

Созданы модели бизнес-процессов использования информационной системы в образовательных целях и редактирования информации «КАК ДОЛЖНО БЫТЬ» с учетом применения в нем разрабатываемого ПО. Разработка ИС распланирована, сроки выполнения отображены в диаграмме Ганта.

Глава 2 Логическое проектирование информационной системы для музея

2.1 Выбор технологии логического моделирования информационной системы для музея

В качестве основного инструмента логического моделирования используется язык UML. UML (Unified Modeling Language) является стандартной нотацией визуального моделирования информационных систем.

Графическая нотация стандарта UML включает в себя широкий набор диаграмм, позволяющих описать требуемые аспекты разрабатываемой системы. Набор диаграмм, используемых при разработке конкретного программного продукта, определяется разработчиком, в зависимости от требований к проекту и необходимого уровня полноты описания. Использование диаграмм нотации UML при разработке является общепринятым решением, поскольку они обеспечивают достаточно подробное описание требуемых спецификаций при сохранении простоты перевода моделей в программный код. В данной работе для общего описания функционала продукта и существующих ролей используется диаграмма вариантов использования; для описания физических особенностей информационной системы диаграмма компонентов; для отображения взаимодействия во времени между модулями системы используется диаграмма последовательности; для моделирования логической схемы базы данных ER-диаграмма; также в главе 3 для описания физической модели развертывания аппаратных компонентов используется диаграмма развертывания [3].

Из таблицы 2 видно, что StarUML имеет преимущество перед другими программными решениями, а также имеет бессрочный период бесплатного использования, поэтому все модели информационной системы построены именно в этой программе.

2.2 Разработка логической модели информационной системы для музея и ее описание

Диаграмма вариантов использования используется для того, чтобы определить, какие актеры пользуются ИС и как они могут с ней взаимодействовать.

Актеры:

- а) школьники, посетители музея, педагоги – потребители информации, которые имеют возможность воспользоваться ИС для получения информации;
- б) разработчик ИС – человек, который имеет доступ к исходному коду ИС;
- в) администратор ИС – человек, который вносит изменения в хранимую в базе данных информацию.

Прецеденты описаны в таблице 3, описание прецедентов в таблицах 4 – 9.

Таблица 3 – Описание прецедентов

ID	Прецедент	Описание
1	Просмотр информации	Действие потребителя информации и администратора ИС для получения данных с использованием ИС
2	Удаление информации	Действие администратора и разработчика ИС для уничтожения информации с базы данных
3	Редактирование информации	Действие администратора и разработчика ИС для внесения изменений в уже существующие записи в базе данных
4	Запись информации	Действие администратора и разработчика ИС для создания записи в базе данных
5	Управление правами доступа	Действие разработчика ИС по созданию новых записей о пользователях в базе данных и редактированию роли, логина и пароля для уже существующих записей в базе данных
6	Редактирование ключевой информации в ИС	Действие разработчика ИС для изменения или создания записей без которых не будут работать те или иные возможности ИС

Таблица 4 – Описание прецедента «Просмотр информации»

Прецедент: Просмотр информации
ИД: 1
Главные актеры: <ol style="list-style-type: none"> 1. школьники; 2. посетители музея; 3. педагоги; 4. администратор ИС.
Основной поток: <ol style="list-style-type: none"> 1. зайти на сайт; 2. выбрать интересующий раздел на сайте; 3. просмотреть информацию.
Альтернативные потоки: <ol style="list-style-type: none"> 1. поток 1: <ol style="list-style-type: none"> a. открыть мобильное приложение; b. выбрать интересующий раздел в мобильном приложении; c. просмотреть информацию. 2. поток 2: <ol style="list-style-type: none"> a. открыть мобильное приложение; b. открыть сканер qr-кода; c. отсканировать qr-код на экспонате; d. просмотреть информацию.

Таблица 5 – Описание прецедента «Удаление информации»

Прецедент: Удаление информации
ИД: 2
Предусловие: актер должен иметь логин и пароль от панели администратора
Главные актеры: <ol style="list-style-type: none"> 1. администратор ИС; 2. разработчик ИС.
Основной поток: <ol style="list-style-type: none"> 1. открыть панель администратора; 2. ввести логин и пароль; 3. выбрать раздел, в котором нужно удалить сущность; 4. выбрать запись, которую надо удалить; 5. нажать кнопку «Удалить».

Таблица 6 – Описание прецедента «Редактирование информации»

Прецедент: Редактирование информации
ИД: 3
Предусловие: актер должен иметь логин и пароль от панели администратора
Главные актеры:

<ol style="list-style-type: none"> 1. администратор ИС; 2. разработчик ИС.
<p>Основной поток:</p> <ol style="list-style-type: none"> 1. открыть панель администратора; 2. ввести логин и пароль; 3. выбрать раздел, в котором нужно удалить сущность; 4. выбрать запись, которую надо удалить; 5. нажать кнопку «Удалить».

Таблица 7 – Описание прецедента «Запись информации»

Прецедент: Запись информации
ID: 4
<p>Предусловие: актер должен иметь логин и пароль от панели администратора</p>
<p>Главные актеры:</p> <ol style="list-style-type: none"> 1. администратор ИС; 2. разработчик ИС.
<p>Основной поток:</p> <ol style="list-style-type: none"> 1. открыть панель администратора; 2. ввести логин и пароль; 3. выбрать раздел, в котором нужно отредактировать сущность; 4. выбрать запись, которую надо отредактировать; 5. написать нужную информацию в поля; 6. нажать кнопку «Сохранить».
<p>Альтернативные потоки:</p> <ol style="list-style-type: none"> 1. поток 1: <ol style="list-style-type: none"> a. открыть панель администратора; b. ввести логин и пароль; c. на главной странице панели администратора выбрать блок, который надо отредактировать; d. написать нужную информацию в поля; e. нажать кнопку «Сохранить».

Таблица 8 – Описание прецедента «Управление правами доступа»

Прецедент: Управление правами доступа
ID: 5
<p>Предусловие: актер должен иметь логин и пароль от панели администратора</p>
<p>Главные актеры:</p> <ol style="list-style-type: none"> 1. разработчик ИС.
<p>Основной поток:</p> <ol style="list-style-type: none"> 1. открыть панель администратора; 2. ввести логин и пароль; 3. выбрать раздел «Пользователи»; 4. выбрать пользователя, которого надо отредактировать; 5. написать нужную информацию в поля; 6. нажать кнопку «Сохранить».

Альтернативные потоки:

1. поток 1:
 - a. открыть панель администратора;
 - b. ввести логин и пароль;
 - c. выбрать раздел «Пользователи»;
 - d. нажать кнопку «Создать»;
 - e. написать нужную информацию в поля;
 - f. нажать кнопку «Сохранить».

Таблица 9 – Описание прецедента «Редактирование ключевой информации в ИС»

Прецедент: Редактирование ключевой информации в ИС
ID: 6
Предусловие: актер должен иметь логин и пароль от панели администратора
Главные актеры: 1. разработчик ИС.
Основной поток: <ol style="list-style-type: none">1. открыть панель администратора;2. ввести логин и пароль;3. выбрать раздел, в котором нужно отредактировать сущность;4. выбрать запись, которую надо отредактировать;5. написать нужную информацию в поля;6. нажать кнопку «Сохранить».

На рисунке 3 изображена диаграмма вариантов использования.

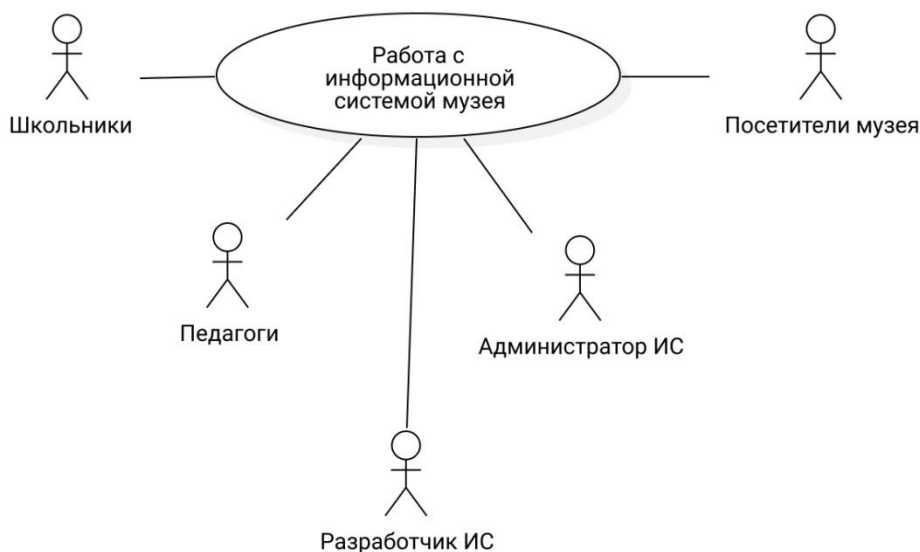


Рисунок 3 – Диаграмма вариантов использования информационной системы

Один общий прецедент не отражает всю функциональность ИС, поэтому выполнена декомпозиция диаграммы вариантов использования (рисунок 4).

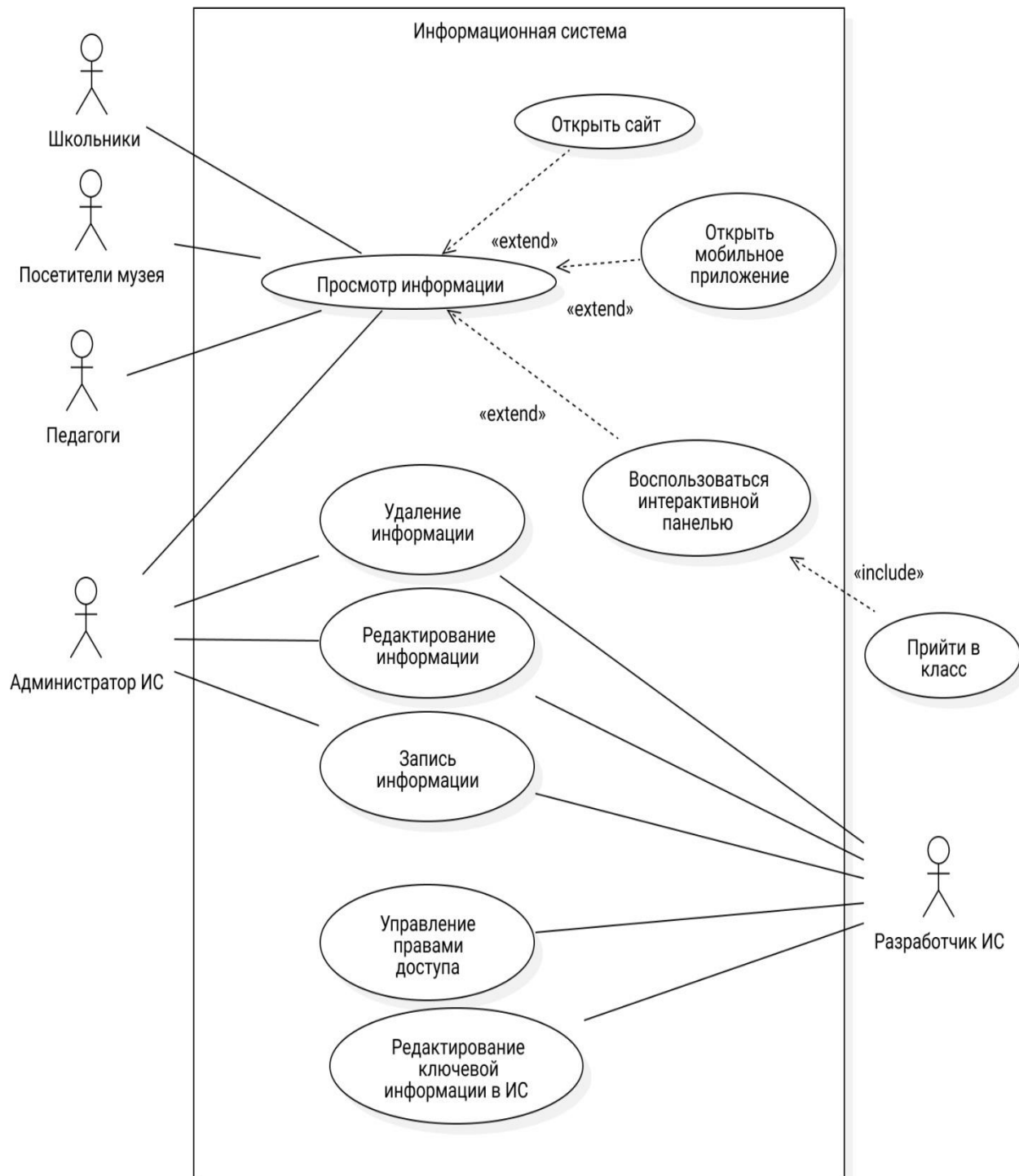


Рисунок 4 – Декомпозиция диаграммы вариантов использования информационной системы

Для того чтобы понять из каких компонент состоит ИС и как они взаимодействуют между собой разработана диаграмма компонентов (рисунок 5).

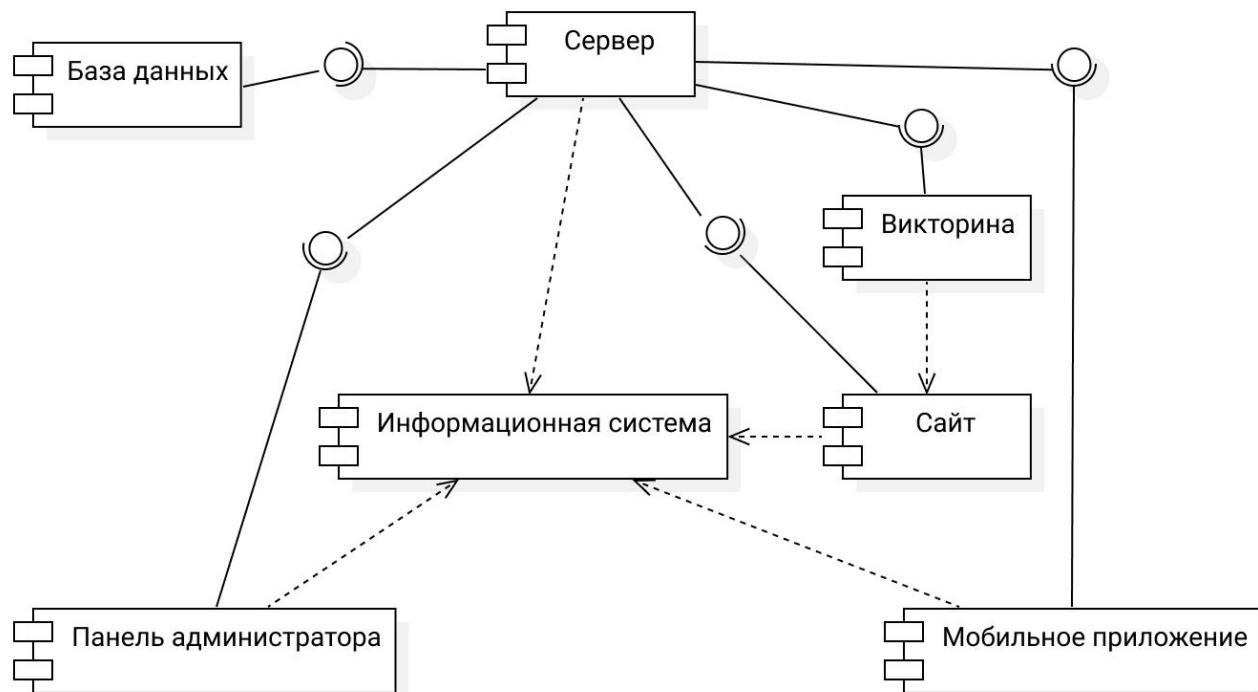


Рисунок 5 – Диаграмма компонентов информационной системы

Из диаграммы видно, что ИС состоит из сервера, сайта, мобильного приложения и панели администратора. Все клиентские составляющие приложения получают информацию на сервере. Сервер отправляет запросы к базе данных для получения информации.

Диаграмма последовательности, отображающая порядок получения данных пользователем, представлена на рисунке 6. Пользователь взаимодействует с приложением, приложение проверяет, есть ли у пользователя права на доступ к информации. Если прав не хватает, то приложение выводит ошибку. Если права доступа есть, то приложение посылает запрос к базе данных на получение информации. База данных в ответном сообщении отправляет информацию приложению. Приложение

формирует из информации понятный пользователю ответ и выводит информацию в интерфейсе.

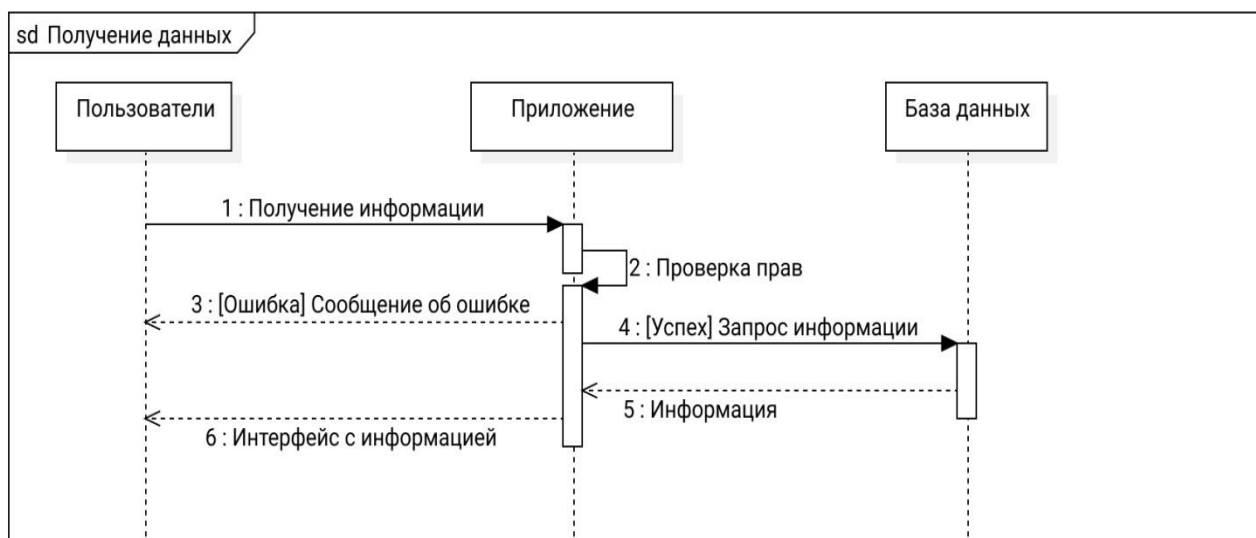


Рисунок 6 – Диаграмма последовательности получения данных

Редактирование, добавление и удаление информации на диаграмме последовательности выглядит одинаково, поэтому эти процессы можно отобразить на одной диаграмме. Диаграмма представлена на рисунке 7.

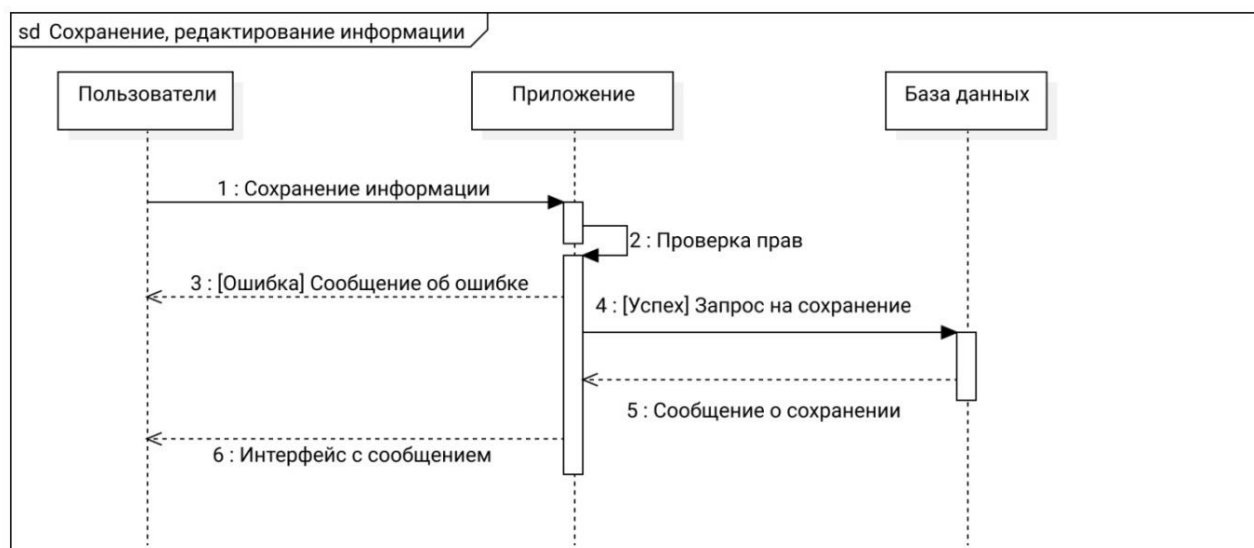


Рисунок 7 – Диаграмма последовательности сохранения данных

Пользователь отправляет запрос на сохранение информации. Приложение проверяет права пользователя. Если у пользователя недостаточно прав, то приложение выводит информацию об ошибке. Если прав достаточно, то приложение отправляет запрос к базе данных. База данных сохраняет данные и отправляет информацию о сохранении данных. Приложение преобразует информацию в сообщение понятное пользователю и отправляет пользователю.

2.3 Проектирование базы данных информационной системы для музея

Наиболее популярные методы для проектирования базы данных IDEF1X и ER-модель в UML.

IDEF1X – это методология для разработки модели реляционных баз данных, она использует специально разработанный синтаксис, который удобен для построения концептуальных моделей баз данных. Так как методология IDEF1X развивается уже достаточно много времени, то она имеет преимущество перед ER-модель из UML, так как имеет строгую спецификацию [7].

Проектирование модели данных в UML только набирает популярность, поэтому UML не имеет спецификаций для моделирования данных. ER-модель может быть использована вместо диаграммы классов в методологии UML. Для разработки данного проекта не используются объектно-ориентированные языка программирования, поэтому для отображения сущностей базы данных использована ER-модель. Она может заменить диаграмму классов и отобразить структуру базы данных.

Для описания объектов виртуальной базы данных составлена логическая модель данных информационной системы (рисунок 8).

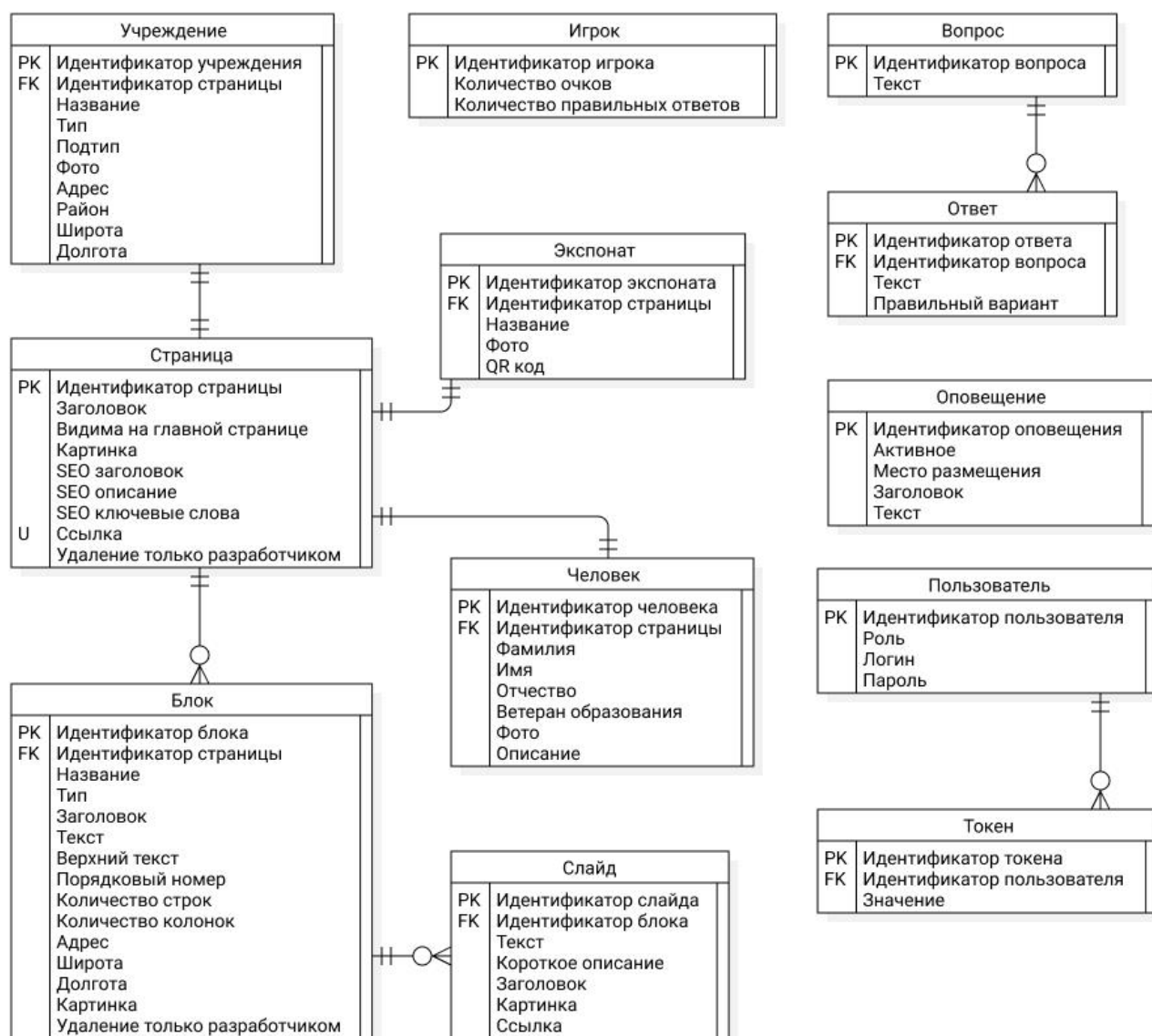


Рисунок 8 – Логическая модель данных информационной системы

Логическая модель содержит 12 сущностей. Каждая сущность имеет уникальный идентификатор, по которому связывается с другими сущностями.

Сущность «Страница» связана с сущностью «Блок» связью «один-ко-многим», а с сущностями «Учреждение», «Экспонат», «Человек» связью «один-к-одному». Это обусловлено тем, что на странице может быть много блоков, но к «Учреждению», «Экспонату» и «Человеку» привязана только одна страница. Сущность «Страница» имеет уникальное поле «Ссылка» - это поле, которое задает часть URL адреса страницы. Каждый блок страницы

может состоять из нескольких однотипных элементов, которые в базе данных хранятся в таблице «Слайд». Исходя из того, что слайдов у блока может быть несколько «Блок» и «Слайд» связаны связью «один-ко-многим».

Независимая сущность «Игрок» – это таблица, хранящая информацию о пользователях, прошедших викторину и результаты прохождения викторины.

Независимая сущность «Оповещения» нужна для отправки оповещения на следующие компоненты ИС: мобильное приложение, сайт, панель администратора.

Сущность «Вопрос» связана с сущностью «Ответ» связью «один-ко-многим», так как к одному вопросу привязано несколько вариантов ответа, для создания викторины по типу: выбор правильного ответа из нескольких предложенных.

Сущность «Пользователь» связана с сущностью «Токен» связью «один-ко-многим», так как пользователь может аутентифицироваться на разных устройствах. Пользователь может иметь несколько токенов, чтобы при новой аутентификации не пропадали предыдущие, у которых срок действия токена еще актуальный. Так же при таком подходе, можно отследить время аутентификации пользователя, даже если срок действия токена уже истек.

Выводы по главе 2

В данной главе была построена объектная модель информационной системы – диаграмма вариантов использования. Также была разработана логическая модель данных согласно выбранной методологии. И определены компоненты информационной системы.

Глава 3 Физическое проектирование информационной системы для музея

3.1 Выбор архитектуры для информационной системы для музея

Наиболее популярные архитектуры для реализации ИС: «клиент-сервер», «файл-сервер» [1].

«Клиент-сервер» – самая распространенная архитектура ИС. Представляет собой взаимодействие двух компонент клиентов и серверов, причем в данной архитектуре клиент начинает общение с сервером, а сервер только отвечает на запросы. Клиенты отправляют запросы к серверу для обмена информацией. Клиент может общаться запросами с несколькими серверами. Серверы могут общаться с несколькими клиентами или несколькими серверами. Задача клиентской части состоит во взаимодействии с пользователем. Двухзвенный «клиент-сервер» – архитектура подразумевает, что обработка запросов происходит на одной машине, без использования сторонних серверов [9]. Многозвенный «клиент-сервер» – архитектура подразумевает, что запросы клиента могут обрабатывать несколько серверов, тем самым снижается нагрузка на сервер. В качестве дополнительного сервера может выступать сервер базы данных.

«Файл-сервер» – используют сетевой ресурс для хранения данных [13]. «Файл-серверная» архитектура проста в реализации и подходит для компьютеров разного уровня развитости, поэтому до сих пор пользуется популярностью. База данных хранится в файл-сервере и для каждого клиента создается копия базы данных, что сказывается на работе «файл-серверной» архитектуры при большом числе клиентов – если их много, то сервер будет работать медленно или вообще не справляться с обработкой запросов.

Сравнение архитектур приведено в таблице 10.

Таблица 10 – Сравнительная характеристика архитектур информационных систем

	Двухзвенный «клиент-сервер»	Многозвенный «клиент-сервер»	«Файл-сервер»
Многопользовательский режим работы с данными	+	+	+
Централизованное управление	+	+	+
Низкая стоимость разработки	-	-	+
Высокая скорость разработки	+	+	+
Легкое подключение новых клиентов	+	+	-
Надежность системы	+	+	-
Высокая производительность	+	+	-
Распределение функций вычислительной системы	-	+	
Итого	6	7	4

Из сравнительной таблицы видно, что для реализации ИС подходит многозвенный «клиент-сервер», так как он набрал больше положительных характеристик, которые необходимо учесть при разработке. Для реализации ИС для музея выбрана многозвенная «клиент-серверная» архитектура как наиболее подходящая для ИС.

3.2 Выбор технологии разработки программного обеспечения для информационной системы для музея

Так как выбрана многозвенная «клиент-серверная» архитектура, то ИС должна состоят из клиентской части, сервера для обработки запросов от клиента и сервера базы данных.

Для реализации сервера для обработки запросов от клиента выбрана платформа Node.js. Платформа предназначена для построения масштабируемых сетевых приложений. При каждом соединении клиента с сервером, запускается функция обратного вызова. Платформа имеет методы

для работы с файловой системой в двух вариантах – блокирующие и не блокирующие [15].

Существует несколько вариантов разработки клиентской части сайта.

HTML, CSS, JS – стандартный набор средств для разработки клиентской части сайта, используется во всех перечисленных способах разработки, а так же может использоваться самостоятельно. Однако использование только этих трех технологий в большой ИС очень долго по времени. Сайты, написанные только с использованием этих технологий, сложно масштабировать, исходных файлов получается много, а размер файлов зачастую превышает 1000 строк кода, в которых сложно разобраться. Разработка таким способом не подразумевает деление сайта на компоненты, поэтому код будет повторяться и для одинаковых элементов сайта надо писать код каждый раз, когда этот элемент используется.

Шаблонизаторы – этот способ разработки подразумевает отрисовку HTML файлов на стороне сервера. Одну страницу сайта можно разбить на несколько шаблонов, а серверная логика соберет их в одну целую страницу. При использовании шаблонов так же можно использовать деление CSS и JS файла на несколько маленьких. При использовании шаблонов страница сайта загружается быстрее и не нагружает устройство пользователя, так как клиент получает готовый HTML файл.

SPA (single page application) – одностраничные веб-приложения. Способ разработки сайтов, при котором весь сайт – это один HTML файл, элементы на сайт добавляются за счет скриптов. Этот способ разработки позволяет изменять на сайте только те элементы, которые должны измениться, все остальные останутся неизменными, это позволяет визуально менять пользователю страницу без ее перезагрузки. Минусом данного способа разработки является плохая поисковая оптимизация, так как все страницы сайта это один HTML файл, созданный заранее, но на нем нет элементов специфичных для каждой страницы, поэтому поисковые системы не понимают какое содержимое у страницы и не включают страницу в

поисковую выдачу. Чтобы избежать такой проблемы страницы можно отрисовывать на сервере, то есть создать гибрид шаблонизатора и SPA. SPA обычно создаются с использованием уже готовых библиотек или фреймворков, таких как React, Angular, Vue.

PWA (progressive web application) – прогрессивные веб-приложения. Технология web-разработки, с помощью которой можно трансформировать сайт в мобильное приложение. Сайты, разработанные на этой технологии, могут отправлять уведомления в браузере или на смартфон, взаимодействовать с аппаратным обеспечением устройства.

Для панели администратора выбрано SPA, так как панель администратора подразумевает много взаимодействия с пользователем. Метод одностраничных приложений позволит создать положительный пользовательский опыт от использования панели администратора.

Существует три основные и популярные технологии для разработки SPA: библиотека React, фреймворк Angular, фреймворк Vue.js.

Для выбора технологии проведено сравнение по критериям: мировой опыт использования, модель, язык программирования, технологические возможности.

Характеристика по этим параметрам представлена в таблице 11.

Таблица 11 – Сравнительная характеристика технологий для разработки SPA

Критерий	React	Angular	Vue.js
Мировой опыт использования	1 место по скачиванию	3 место по скачиванию	2 место по скачиванию
Модель	На основе виртуального DOM	MVC	На основе виртуального DOM
Язык программирования	JavaScript	TypeScript	JavaScript, HTML
Технологические возможности	Разработка веб-приложений и нативных приложений[20].	Разработка крупномасштабных и многофункциональных приложений [6].	Разработка легковесных веб-приложений [21].

ИС не попадает под разряд крупномасштабной, поэтому Angular избыточен для реализации ИС, к тому же Angular использует язык программирования TypeScript и порог вхождения в разработку на Angular повышается [16]. React и Vue.js имеют схожие параметры, но Vue.js может быть использован без дополнительных библиотек, так как все необходимое есть в ядре технологии. React из всех исследуемых технологий является самым гибким и технологии, которые используются с ним выбираются разработчиком. Благодаря этому React имеет много совместимых с ним библиотек и широкий опыт мирового использования.

Для разработки SPA выбрана библиотека React, как наиболее популярное и гибкое решение. В панели администратора используется дополнительный набор компонентов из библиотеки react-admin – это набор уже готовых компонентов веб-приложения для создания форм, аналитики.

3.3 Выбор системы управления базой данных для информационной системы для музея

Так как ИС имеет много сущностей и предполагает масштабирование, то выбрана модель реляционной базы данных. Для разработки ИС выбрана СУБД PostgreSQL. Выбор в пользу PostgreSQL был сделан исходя из опыта работы с ней и опытом использования этой СУБД в мировой практике.

PostgreSQL плюсы:

- максимальный размер базы данных не ограничен;
- надежные и высокопроизводительные системы репликации и транзакций;
- расширяемость;
- свободное распространение;
- поддержка форматов данных, необходимых в ИС;
- в системе встроенных языков есть движок V8 для JavaScript.

Утилиты для СУБД позволяют создавать резервные копии, как через интерфейс, так и через командную строку.

Еще одним плюсом в пользу PostgreSQL является возможность бесплатно развернуть базу данных на бесплатном хостинге Heroku, который использован для демонстрации рабочей версии ИС заказчику. Хостинг имеет набор команд для работы с базой данных из терминала CMD.

3.4 Разработка физической модели данных для информационной системы для музея

Для того чтобы понять какие сущности и поля должны быть в базе данных требуется физическая модель данных. Модель изображена на рисунке 9.

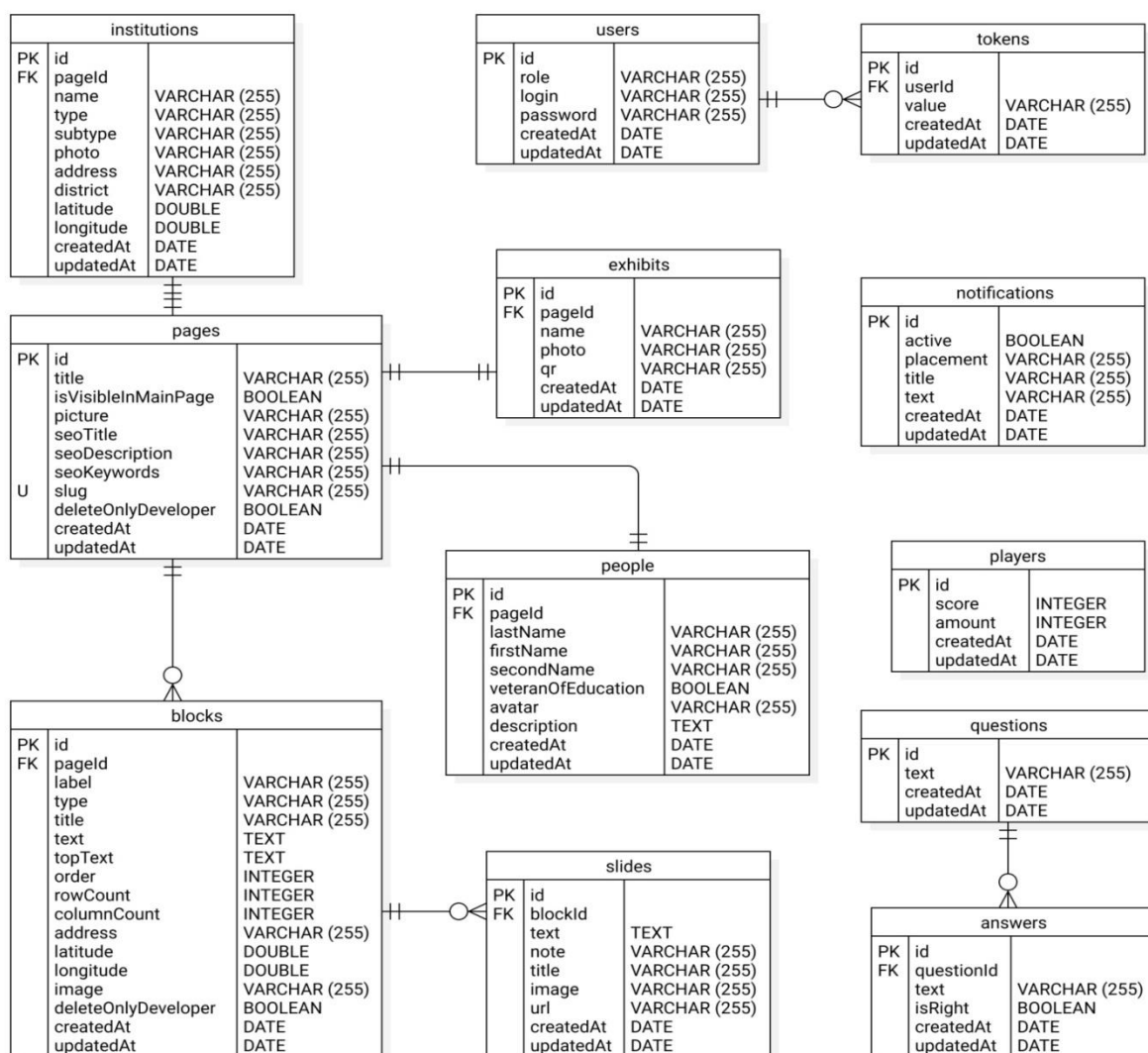


Рисунок 9 – Физическая модель базы данных для информационной системы

Физическая модель базы данных является последним этапом проектирования, на ней показана вся информация необходимая для разработки.

3.5 Разработка программного обеспечения информационной системы для музея

3.5.1 Разработка базы данных

Использование реляционной базы данных приводит к семантическому разрыву, поэтому требуется писать программное обеспечение (ПО) для обработки данных в объектно-ориентированном виде и ПО для сохранения данных в реляционном виде. Для создания базы данных используется ORM система Sequelize. ORM система решает проблему преобразования данных между двумя формами данных, тем самым повышая производительность программиста и снижая сложность преобразования данных [18]. Конфигурация базы данных представлена на листинге 1.

Листинг 1 – Конфигурация базы данных с помощью ORM Sequelize

```
const Sequelize = require('sequelize');
const sequelize = new Sequelize(process.env.DB_URL,
{
  logging: false,
  rejectUnauthorized: true,
  }
);
module.exports = sequelize;
```

Из переменных среды окружения берется ссылка на базу данных, содержащая ее название, имя пользователя, пароль, порт, адрес до сервера, на котором размещена база данных. Для создания сущностей в базе данных используются файлы с описанием моделей. Каждая сущность в базе данных создается при помощи класса, который наследуется от класса Model. Для создания полей сущности используется статический метод `init`. Метод принимает атрибуты и их опции. Для создания связи между сущностями используются статические методы класса Model.

Просмотреть информацию о базе данных и о содержащихся в ней записях можно через утилиту pgAdmin, здесь же можно внести изменения, используя SQL запрос или графический интерфейс (рисунок 10).

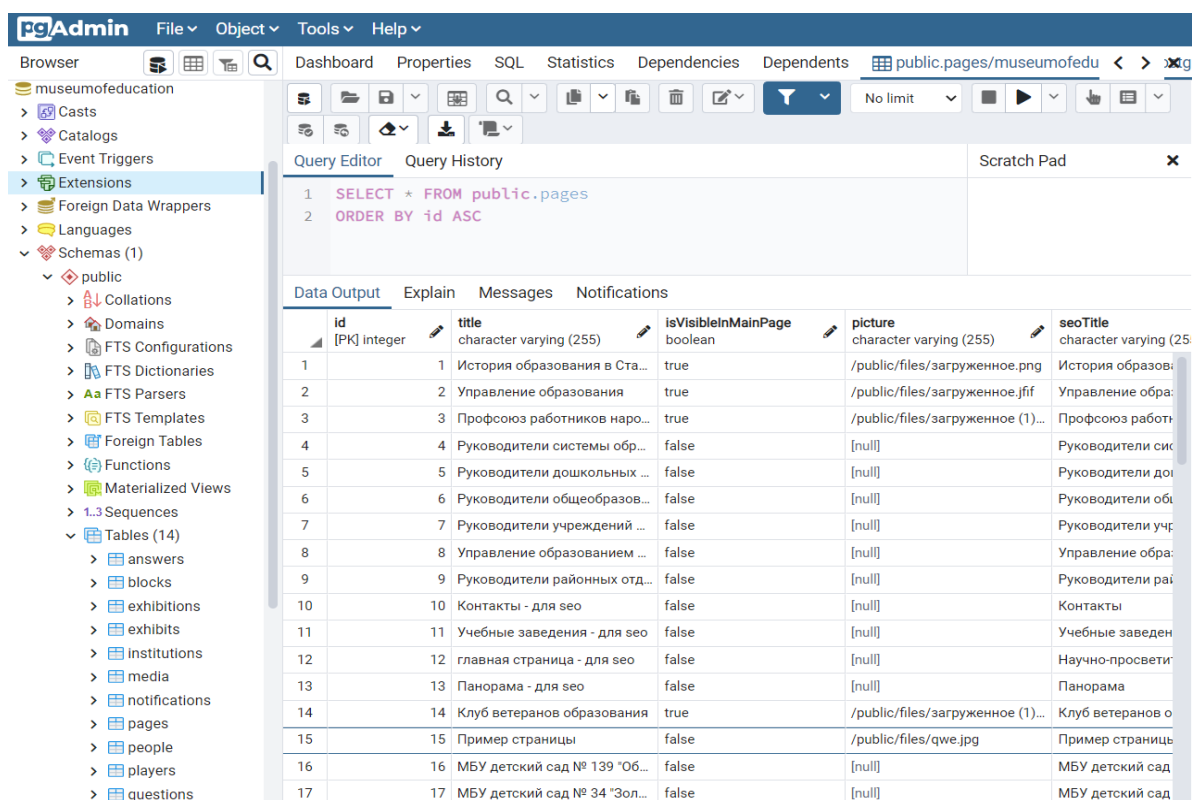


Рисунок 10 – Представление базы данных ИС в утилите pgAdmin

Таким образом, без использования SQLзапросов созданы таблицы и связи между ними. Для получения данных из базы данных в ORM Sequelize существуют методы классов моделей.

3.5.2 Разработка сервера

Серверная часть информационной системы разработана на технологии Node.js. Для упрощения создания серверной логики был использован фреймворк Express. В нем уже встроены функции для обработки запросов, работа с файлами cookies, CORS, раздача статических файлов. Код создания сервера приведен на листинге Д.1. Из листинга видно, что для создания сервера потребовалось всего несколько строк. Следующие функции по созданию сервера выполнил фреймворк Express:

- установил движок для шаблонизатора;
- установил совместное использование ресурсов между разными источниками;
- установил максимальный размер файла, который может быть обработан сервером;
- создал точки для обработки запросов к серверу.

Запуск сервера осуществляется с помощью диспетчера процессов pm2. Он позволяет задать конфигурацию сервера для различных вариантов запуска. Конфигурация к нему задана в файле ecosystem.config.js. В конфигурации задано две группы параметров для запуска сервера. Для запуска в режиме разработки используются параметры «dev», для запуска на выделенных серверах Heroku используются параметры «heroku».

Развертывание сервера на Node.js производится с помощью пакетного менеджера Yarn. Он использует файл конфигурации package.json для установки библиотек, выполнения команд, сборки проекта. В листинге Е.1 представлен код конфигурации файла package.json. Исходя из конфигурации видно, что при создании серверной части ИС были использованы библиотеки:

- @babel/core – версия 7.12.3;
- body-parser – последняя версия;
- browser-sync – версия 2.26.13;

- cors – версия 2.8.5;
- ejs – версия 3.1.5;
- express – версия 4.17.1;
- form-data – версия 3.0.0;
- html-pdf – версия 2.2.0;
- jsdoc – версия 3.6.6;
- jsonwebtoken – версия 8.5.1;
- multer – версия 1.4.2;
- node-fetch – версия 2.6.1;
- node-sass – версия 5.0.0;
- object-to-formdata – версия 4.1.0;
- pg – версия 8.3.3;
- pm2 – версия 4.4.1;
- qrcode – версия 1.4.4;
- request – версия 2.88.2;
- sequelize – версия 6.3.5;
- winston – версия 3.3.3.

Так как сервер выполняет еще и функцию компиляции каскадной таблицы стилей из препроцессора SCSS в CSS, то в списке необходимых библиотек есть библиотеки для компиляции и минификации кода. Для минификации и транспилеризации JavaScript кода используется библиотека Babel [10].

Каждая сущность имеет примерно одинаковый набор запросов, поэтому во избежание дублирования кода написаны функции для:

- получения одной записи;
- получения списка с возможностями фильтрации и сортировки;
- удаления одной записи;
- редактирования одной записи;
- получения 10 случайных записей;

- получения правильных ответов;
- создания записи.

Все точки для запросов имеют промежуточный слой для проверки прав пользователя.

Важной частью при разработке сервера было написание функций для получения данных из существующей ИС, для этого были написаны запросы с использованием библиотеки `node-fetch`.

3.5.3 Разработка клиента

Важной частью ИС является панель администратора, так как благодаря ей администратор может пополнять базу данных новыми записями и управлять существующей информацией. Реализация панели администратора велась с использованием библиотеки `react`. Для создания проекта для разработки панели администратора использовался стандартный проект – CRA. CRA (`create react application`) – способ для начала создания одностраничного приложения на React. В нем создана структура директорий, которая позволила хранить части приложения и легко ориентироваться между файлами.

При реализации панели администратора в дополнение к библиотеке `react` использовано b2b решение `react-admin`. Набор готовых интерактивных элементов интерфейса позволил создать формы для манипуляции с записями базы данных. Вся логика работы веб-приложения заключена в готовом компоненте `Admin`. Аргументами в него передаются функции для аутентификации и авторизации, функции для отправки запросов к базе данных. С помощью возвращаемого компонентом аргумента `permissions` можно проверить роль пользователя и отобразить необходимые для него компоненты. Все необходимые поля для ввода уже содержатся в библиотеке, поэтому интерфейс для заполнения информации был собран из готовых компонентов. Библиотека так же позволяет добавить возможность создания одной сущности из другой. Такой функционал может быть полезен,

например, при создании вопросов для викторины. При их создании удобно сразу создавать ответы.

Используемые библиотеки и команды для запуска сервера для разработки и сборки веб-приложения содержатся в файле `package.json`. Установка и использование команд осуществляется посредством файлового менеджера `Yarn`, так же, как и для серверной части ИС.

Для упрощения работы с панелью администратора на главной странице написана инструкция по работе, вынесены кнопки для редактирования ключевых блоков (рисунок 11).

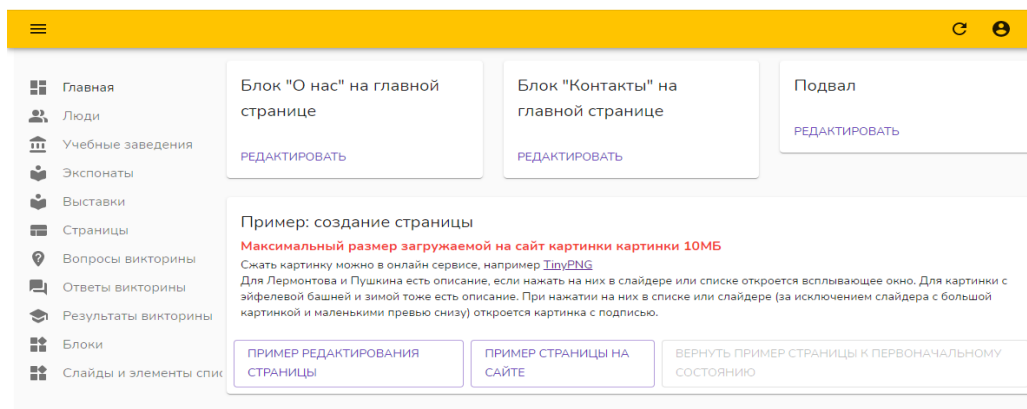


Рисунок 11 – Главная страница панели администратора

В панели администратора отображаются все сущности из базы данных, которые администратор может редактировать. У разработчика в панели администратора отображается больше информации, это служебная информация, необходимая для оценки правильности работы ИС и выявления ошибок. Пример отображения записей показан на рисунке 12.

id ↑	Название	Тип	Подтип
1	МБУ детский сад № 139 "Облачко"	Дошкольное образовательное учреждение	РЕДАКТИРОВАТЬ
2	МБУ детский сад № 34 "Золотая рыбка"	Дошкольное образовательное учреждение	РЕДАКТИРОВАТЬ
3	МБУ детский сад № 147 "Сосенка"	Дошкольное образовательное учреждение	РЕДАКТИРОВАТЬ
4	МБУ детский сад № 46 "Игрушка"	Дошкольное образовательное учреждение	РЕДАКТИРОВАТЬ
5	МАОУ детский сад № 79 "Гусельки"	Дошкольное образовательное учреждение	РЕДАКТИРОВАТЬ
6	МАОУ ДС № 80 «Песенка»	Дошкольное образовательное учреждение	РЕДАКТИРОВАТЬ
7	МБУ детский сад № 90 "Золотое зернышко"	Дошкольное образовательное учреждение	РЕДАКТИРОВАТЬ
8	МБУ детский сад № 196 "Маячок"	Дошкольное образовательное учреждение	РЕДАКТИРОВАТЬ
9	МБУ "Школа № 10"	Школа	РЕДАКТИРОВАТЬ
10	МБУ "Школа № 16"	Школа	РЕДАКТИРОВАТЬ

Строк на странице: 10 | 1-10 из 30 | 1 | 2 | 3 | [СЛЕДУЮЩАЯ >](#)

Рисунок 12 – Раздел со списком учебных заведений

При нажатии на запись или кнопку «Редактировать» откроется больше информации и ее можно будет отредактировать при необходимости (рисунок 13). Сущности «Учебные заведения», «Экспонаты», «Люди» связываются со страницей, чтобы можно было добавлять более развернутую информацию. Привязать страницу к этим сущностям можно в панели администратора в отдельной вкладке.

Для роли «Разработчик» в панели администратора имеется расширенный функционал. Разработчик видит и может редактировать скрытые от администратора поля. У администратора нет прав для редактирования этих полей, так как это может привести к сбою работы ИС. У разработчика есть возможность создавать оповещения, для уведомления администратора об изменениях в панели администратора (рисунок 14).

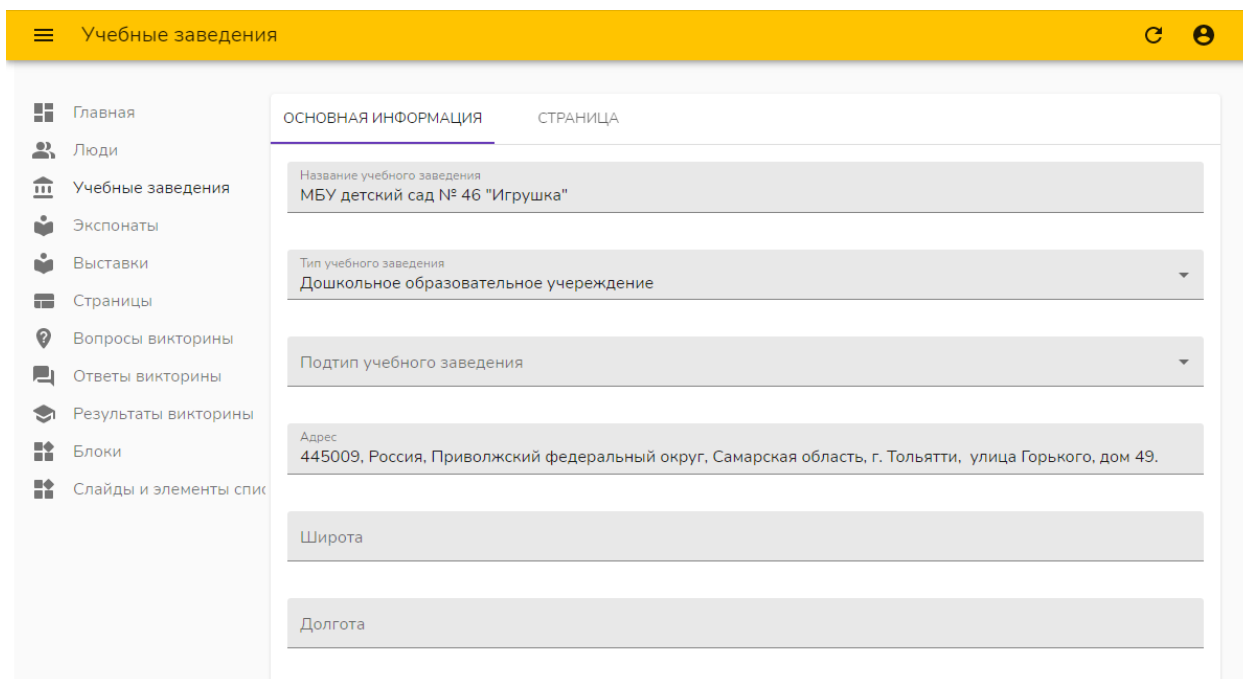


Рисунок 13 – Редактирование учебного заведения

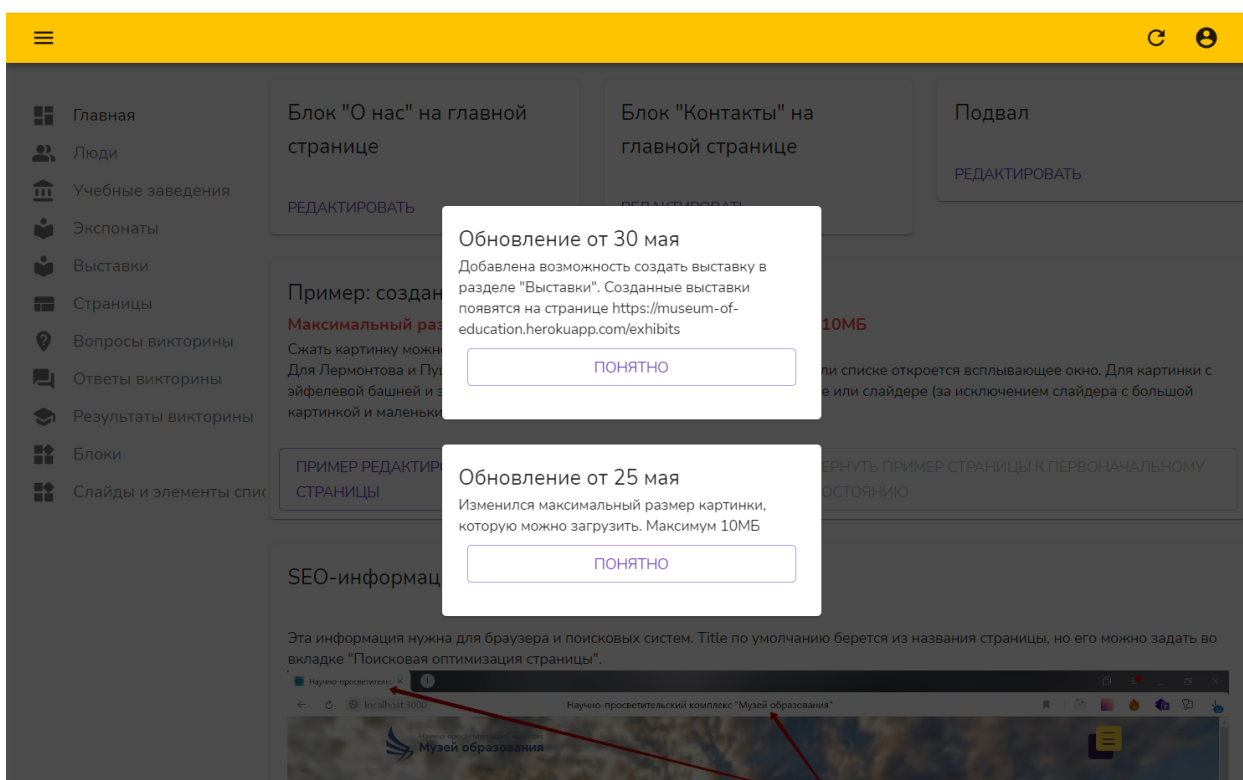


Рисунок 14 – Оповещения в панели администратора

После первого просмотра уведомления, оно больше не будет показано пользователю, если он зайдет с этого же браузера.

3.6 Разработка диаграммы развертывания информационной системы

В ходе выявления требований и разработки ИС получены несколько компонентов, которые должны работать как единое целое и взаимодействовать в соответствии с диаграммой компонентов. Для того чтобы определить расположение компонентов при развертывании на реальном сервере составлена диаграмма развертывания (рисунок 15). Данная диаграмма отображает требования к аппаратно-программному обеспечению.

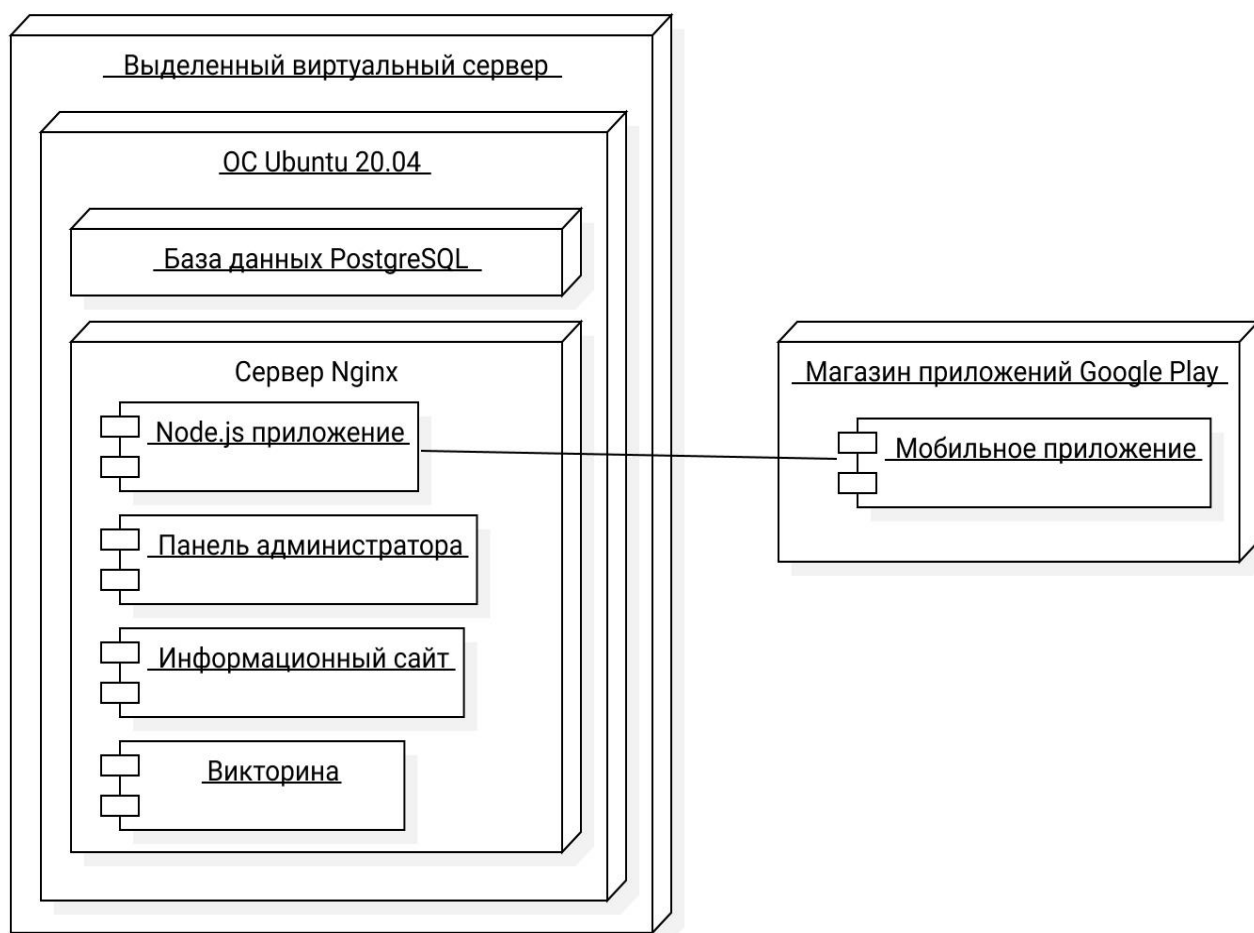


Рисунок 15 – Диаграмма развертывания информационной системы

Из диаграммы видно два основных узла – выделенный виртуальный сервер и магазин приложений GooglePlay.

В узле «Выделенный виртуальный сервер» находится узел с операционной системой Ubuntu 20.04. На операционной системе установлена база данных PostgreSQL и сервер Nginx. На сервере запускается Node.js приложение, также располагаются файлы для панели администратора, сайта и викторины.

В узле «Магазин приложений GooglePlay» находится компонент «Мобильное приложение». Она взаимодействует с компонентом в серверном узле для получения данных из базы данных.

3.7 Тестирование информационной системы

3.7.1 Критерии для тестирования компонентов информационной системы для музея

Для тестирования ИС был выбран способ ручного тестирования. Тестирование производилось методом «черного ящика», а там где требовалось проверить реакцию ИС на сбои, использовался метод «серого ящика». Выбрано тестирование с помощью метода «черного ящика», так как при таком способе тестирование производится с позиции конечного пользователя. Тестировать работу сервера отдельно от клиентской части затратно по времени и не несет практической значимости, так как все компоненты ИС используются вкуче. Во время тестирования проверялась правильная работа ИС как на клиентской части, так и на серверной.

Критерии для тестирования:

- работа ограничений по отображению элементов интерфейса и возможность действий;
- сохранение информации в базу данных;
- правильное отображение данных из базы данных;

- генерация qr-кодов для сущности «Экспонаты»;
- возможность сгенерировать и вывести на печать qr-коды для экспонатов;
- правильная работа сортировок для данных;
- логирование действий пользователя в панели администратора;
- возможность просмотра файлов логирования.

Тест кейсы, которые использовались во время тестирования, описаны в таблице Ж.1.

3.7.2 Тестирование компонентов информационной системы для музея

Сводная таблица результатов тестирования представлена ниже (таблица 12).

Таблица 12 – Результаты тестирования

Критерий	Результат
Работа ограничений по отображению элементов интерфейса и возможность действий.	Корректная работа
Сохранение информации в базу данных.	Данные связанных сущностей не сохраняются в базу данных. Не редактируется картинка.
Правильное отображение данных из базы данных.	Ошибка с постраничным отображением данных
Генерация qr-кодов для сущности «Экспонаты».	Корректная работа
Возможность сгенерировать и вывести на печать qr-коды для экспонатов.	Корректная работа
Правильная работа сортировок для данных.	Корректная работа
Логирование действий пользователя в панели администратора.	Корректная работа
Возможность просмотра файлов логирования.	Корректная работа

Ограничение прав доступа в панели администратора присутствует в нескольких местах – при аутентификации пользователя, при удалении страницы и блока. Действия пользователя ограничивает не только интерфейс, но и запрет с сервера. При проверке работы этого функционала, проблем не обнаружено.

Сохранение и редактирование данных протестировано для всех сущностей. Для проверки правильности сохранения проверялись записи в

базе данных через утилиту pgAdmin. При создании сущностей с зависимыми сущностями выявлены ошибки в сохранении зависимых сущностей. При редактировании данных выявлена ошибка – не сохранение обновленных данных.

Для генерации qr-кодов есть два метода:

- генерация при создании записи об экспонате;
- генерация по нажатию на кнопку в панели администратора.

При тестировании этих методов ошибок не обнаружено.

Для печати qr-кодов в панели администратора можно выбрать нужные и вывести на печать. При тестировании этого функционала ошибок не обнаружено.

Сортировка данных реализована на серверной части с использованием методов ORM Sequelize, а на клиентской части в панели администратора использованы функции из документации react-admin для отправки запросов. Тем самым минимизированы ошибки в работе проверяемого функционала и в результате тестирования ошибки не обнаружены.

Действия панели администратора логируются в файлы app.log и error.log. В файл app.log записывается вся информация о действиях, на которые закреплен слушатель. В файл error.log записываются ошибки во время работы сервера. Ошибки работы сервера минимизированы за счет обработчиков исключений, поэтому для тестирования они были смоделированы специально. В ходе тестирования ошибок не обнаружено.

Выводы по главе 3

Проведено физическое проектирование ИС, а именно:

- выбрана архитектура ИС – многозвенный «клиент-сервер»;

- выбраны технологии для разработки программного обеспечения, для сервера платформа Node.js и фреймворк Express, для клиента библиотека React, B2B решение ReactAdmin;
- выбрана СУБД – PostgreSQL;
- описана разработка программного обеспечения;
- проведено ручное тестирование всех компонент ИС.

На этом проектирование и разработка первой версии ИС завершилась. Получилось клиент-серверное приложение готовое к работе. ИС загружена на бесплатный хостинг Heroku для демонстрации заказчику и проведения тестирования.

В ходе тестов были обнаружены некритические ошибки в работе, все они исправлены, ИС введена в использование.

Заключение

В ходе выполнения выпускной квалификационной работы разработана информационная система для музея.

Во время выполнения работы проведен анализ процессов в организации, поставлена задача на разработку новой информационной системы и составлены требования в системе FURPS+. Для удобства разработки составлены логические модели информационной системы, которые в полной мере описывают необходимый функционал. Информационная система разработана с применением современных средств разработки, таких как Node.js, React. Все выбранные методы для создания сервера и клиентской части используют язык JavaScript, отсюда можно сделать вывод, что информационная система имеет изоморфный код и вполне может поддерживаться 1 программистом.

Данные с предыдущей информационной системы не потерялись, благодаря тому, что на сервере реализован функционал для их получения, обработки в нужную форму и сохранение в новую базу данных.

В панели администратора у администратора появилась возможность гибко изменять отображаемую информацию, а разработчик может через панель администратора изменять ключевую информацию без вмешательства в исходный код ИС. Использование готового b2b решения ReactAdmin ускорило процесс реализации панели администратора и позволило минимизировать ошибки при работе ИС. Администратор имеет возможность ознакомиться с инструкцией по работе в панели администратора и посмотреть примеры заполнения информации.

Реализованная ИС протестирована на описанных тест-кейсах, а выявленные ошибки в работе исправлены. Для проверки ИС заказчиком, она загружена на бесплатный сервер от сервиса Heroku. Заказчиком проверен функционал новой ИС и составлены пожелания для улучшения. После проверки заказчиком ИС введена в использование.

Список используемой литературы

1. Архитектурные особенности проектирования и разработки Веб-приложений [Электронный ресурс]: сайт НОУ ИНТУТ. URL: <https://intuit.ru/studies/courses/611/467/lecture/28784> (дата обращения: 01.04.2021).
2. Выбор и обоснование средств моделирования [Электронный ресурс]: Сайт архив студенческих работ. URL: https://vuzlit.ru/2277071/vybor_obosnovanie_sredstv_modelirovaniya (дата обращения: 01.04.2021).
3. Диаграмма компонентов (component diagram) [Электронный ресурс]: учеб. пособие. URL: http://imlearning.ru/netcat_files/file/FSIS/ФСИС_семинар-9_Диаграмма-компонентов.pdf (дата обращения: 01.04.2021).
4. Дэвид А. Марка и Клемент Мак Гоуэн, Предисловие Дугласа Т. Росса, МЕТОДОЛОГИЯ СТРУКТУРНОГО АНАЛИЗА И ПРОЕКТИРОВАНИЯ SADT // учеб. пособие. С. 17. URL: <https://pqm-online.com/assets/files/lib/books/marka.pdf> (дата обращения: 01.04.2021).
5. Жизненный цикл информационных систем [Электронный ресурс]: Сайт ТГУ. URL: https://edu.tltsu.ru/sites/sites_content/site216/html/media67140/lec2_is-2.pdf (дата обращения: 01.04.2021).
6. Какой фреймворк нужно изучать в 2020 году: Angular, React, или Vue.js [Электронный ресурс]: статья. URL: <https://badcode.ru/chto-luchshie-vsiegno-izuchat-v-2020-ghodu-angular-react-ili-vue-js/> (дата обращения: 01.04.2021).
7. Михаил Кривошеин, ER: диаграммы сущность – связь [Электронный ресурс]: учеб. статья. URL: https://dl.sumdu.edu.ua/drafts/2629/523503/ER_Modeling.pdf (дата обращения: 01.04.2021).

8. Научно-просветительский комплекс «Музей образования» как центр исследовательской и проектной деятельности обучающихся [Электронный ресурс]: Сайт ТолВИКИ. URL:http://wiki.tgl.net.ru/index.php/Музей_образования (дата обращения: 01.04.2021).

9. О модели взаимодействия клиент-сервер простыми словами. Архитектура «клиент-сервер» с примерами [Электронный ресурс]: IT-блог о веб-технологиях, серверах, протоколах, базах данных, СУБД, SQL, компьютерных сетях, языках программирования и создание сайтов. URL: <https://zametkinapolyah.ru/servera-i-protokoly/o-modeli-vzaimodejstviya-klient-server-prostymi-slovami-arxitektura-klient-server-s-primerami.html> (дата обращения: 01.04.2021).

10. Полифилы [Электронный ресурс]: сайт LearnJavaScript. URL: <https://learn.javascript.ru/polyfills> (дата обращения: 01.04.2021).

11. Руководство для начинающих по использованию BPMN в повседневной работе [Электронный ресурс]: Сайт Microsoft. URL: <https://www.microsoft.com/ru-ru/microsoft-365/business-insights-ideas/resources/the-guide-to-using-bpmn-in-your-business> (дата обращения: 01.04.2021).

12. СТРУКТУРА И ОРГАНЫ УПРАВЛЕНИЯ ОБРАЗОВАТЕЛЬНОЙ ОРГАНИЗАЦИЕЙ [Электронный ресурс]: Сайт МБОУ г.о. Тольятти «Школа №4 имени Н.В. Абрамова». URL: <http://school4.tgl.net.ru/struktura-i-organy-upravleniya-obrazovatelnoj-organizatsiej> (дата обращения: 01.04.2021).

13. Типовые архитектуры [Электронный ресурс]: сайт НОУ ИНТУТ. URL: <https://intuit.ru/studies/courses/633/489/lecture/11073> (дата обращения: 01.04.2021).

14. StarUML. Руководство пользователя [Электронный ресурс]: Сайт StarUML. URL: [http://staruml.sourceforge.net/docs/user-guide\(ru\)/user-guide.pdf](http://staruml.sourceforge.net/docs/user-guide(ru)/user-guide.pdf) (дата обращения: 01.04.2021).

15. About Node.js [Электронный ресурс]: сайт Node.js. URL: <https://nodejs.org/en/> (дата обращения: 01.04.2021).

16. Angular [Электронный ресурс]: документация фреймворка Angular. URL: <https://angular.io> (дата обращения: 01.04.2021).

17. INTRODUCTION TO OMG'S UNIFIED MODELING LANGUAGE (UML) [Электронный ресурс]: Сайт uml.org. URL: <https://www.uml.org/what-is-uml.htm> (дата обращения: 01.04.2021).

18. Mapping Objects to Relational Databases: O/R Mapping In Detail [Электронный ресурс]: сайт Agile Data. URL: <http://www.agiledata.org/essays/mappingObjects.html> (дата обращения: 01.04.2021).

19. Microsoft Completes Acquisition of Visio [Электронный ресурс]: Сайт Microsoft. URL: <https://news.microsoft.com/2000/01/07/microsoft-completes-acquisition-of-visio> (дата обращения: 01.04.2021).

20. React. A JavaScript library for building user interfaces [Электронный ресурс]: документация для библиотеки React. URL: <https://reactjs.org> (дата обращения: 01.04.2021).

21. Vue.js [Электронный ресурс]: документация фреймворка Vue. URL: <https://vuejs.org> (дата обращения: 01.04.2021).

Приложение А

Модель «КАК ЕСТЬ» процесса работы с панелью администратора

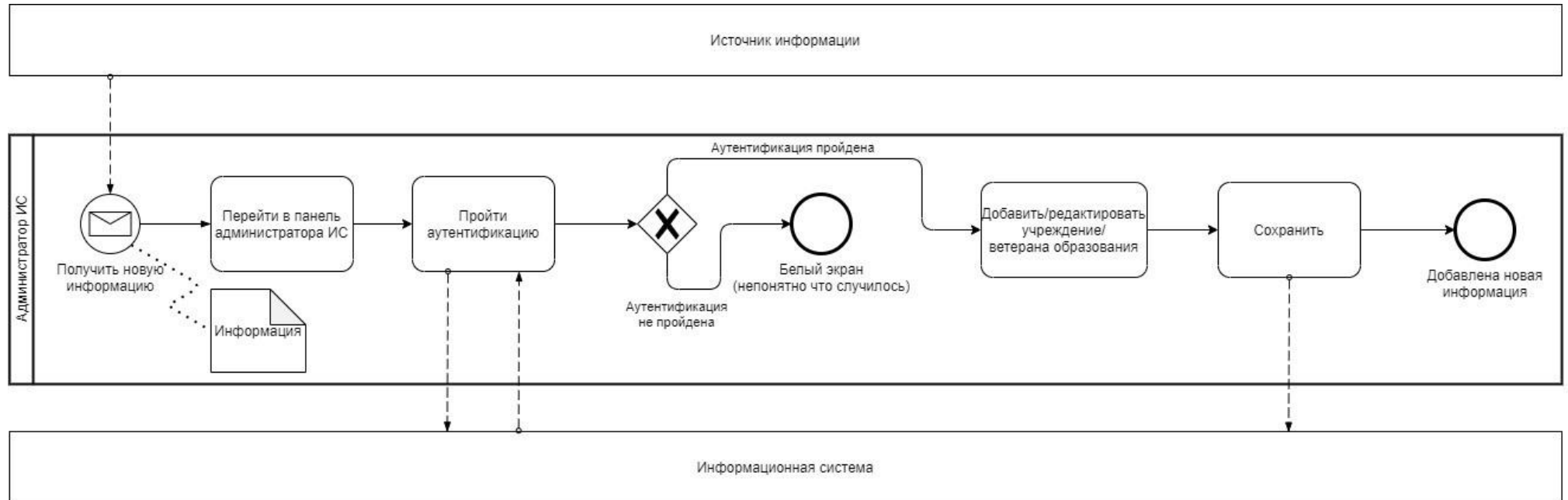


Рисунок А.1 – Модель «КАК ЕСТЬ» процесса работы с панелью администратора

Приложение Б

Модель «КАК ЕСТЬ» процесса использования информационной системы в образовательных целях

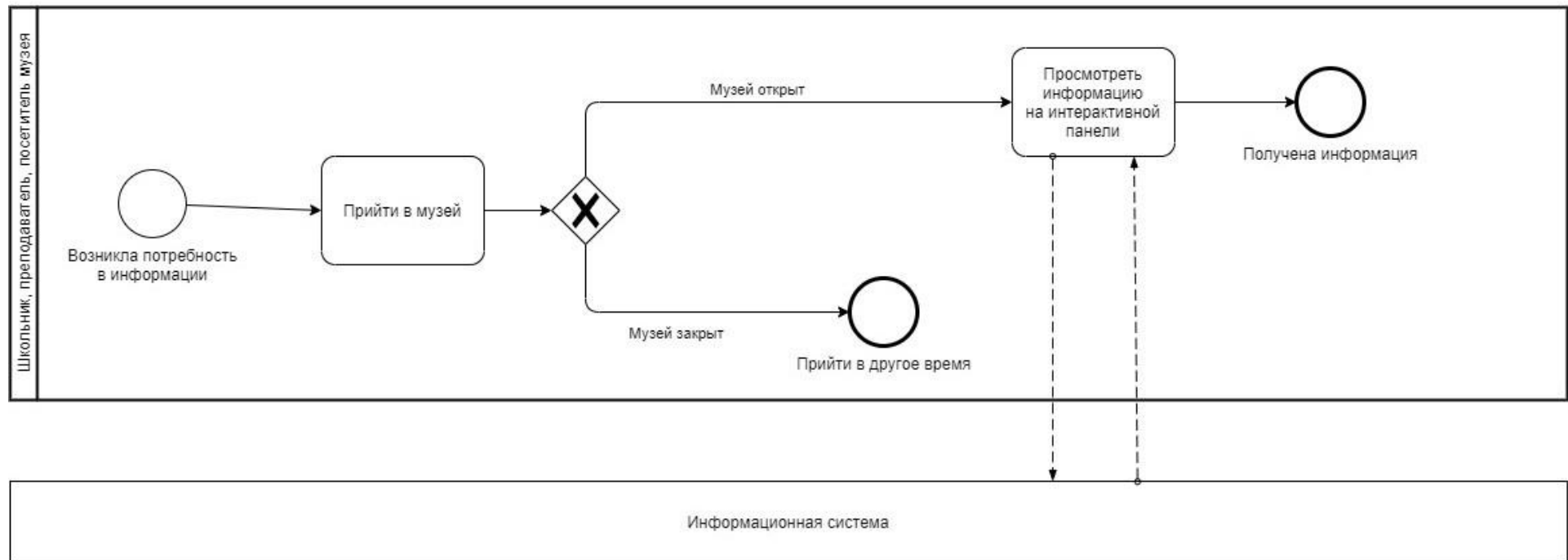


Рисунок Б.1 – Модель «КАК ЕСТЬ» процесса использования информационной системы в образовательных целях

Приложение В

Модель «КАК ДОЛЖНО БЫТЬ» процесса работы с панелью администратора

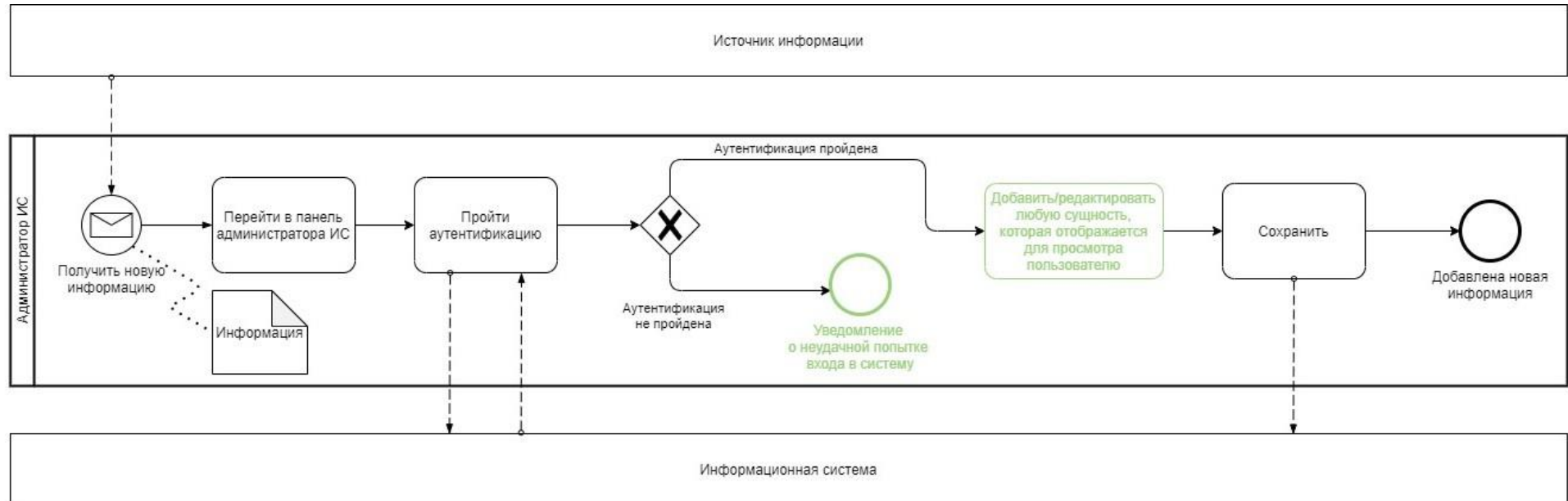


Рисунок В.1 – Модель «КАК ДОЛЖНО БЫТЬ» процесса работы с панелью администратора

Приложение Г

Модель «КАК ДОЛЖНО БЫТЬ» процесса использования информационной системы в образовательных целях

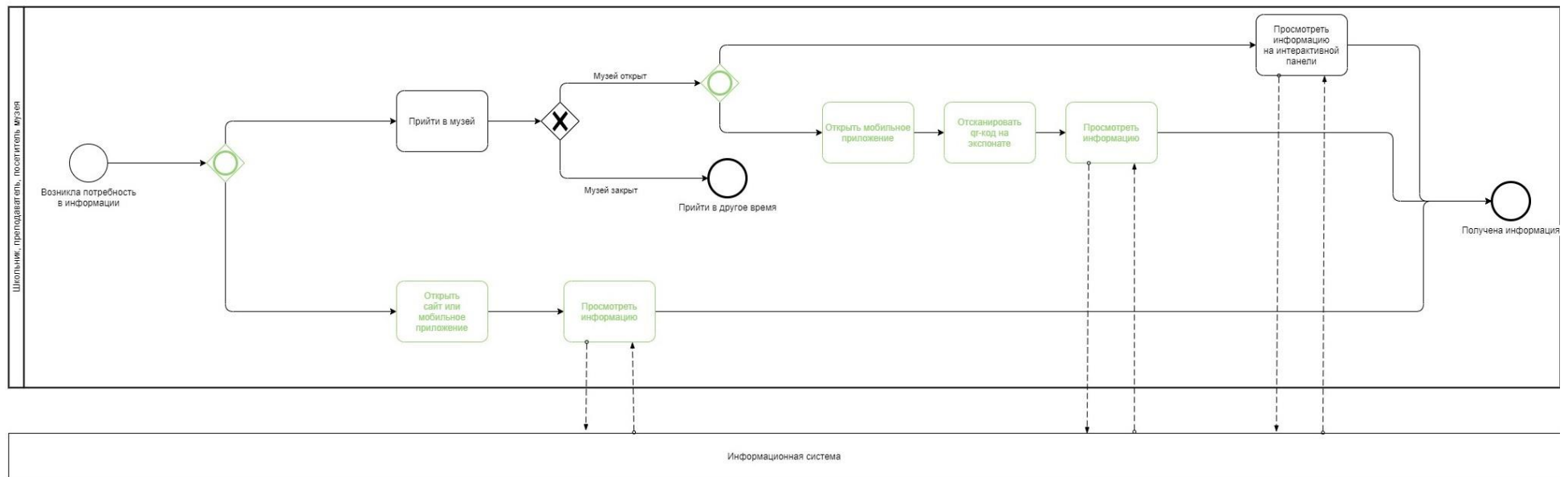


Рисунок Г.1 – Модель Модель «КАК ДОЛЖНО БЫТЬ» процесса использования информационной системы в образовательных целях

Приложение Д

Код создания сервера

Листинг Д.2 – Код создания сервера

```
(async () => {
  try {
    global.rootDir = __dirname;

    const express = require('express');
    const { init } = require('./models');
    const routes = require('./routes');
    const cors = require('cors');
    const bodyParser = require('body-parser');
    const sync = require('./init')
    const renderStyles = require('./libs/renderStyles');
    const concatJS = require('./libs/concatJS');

    await init();
    await sync();

    const app = express();

    app.set("view engine", "ejs");
    app.use(cors());
    app.use(bodyParser.json({limit: '1024Mb'}));
    app.use(routes);
    app.use(async (req, res) => {
      res.status(404).json({_message: 'Страница не найдена'})
    });
  }
});
```

Продолжение Приложения Д

```
await renderStyles();
await concatJS();

app.listen(process.env.PORT || 3000, () => {
  console.log(`Server started at port ${process.env.PORT || 3000}`);
  });
  } catch (e) {
  console.error('Ошибка поднятия сервера', e)
  }
  })());
```

Приложение E

Код из файла конфигурации package.json

Листинг E.3 – Код из файла конфигурации package.json

```
{
  "name": "museum-of-education",
  "version": "1.0.0",
  "main": "index.js",
  "license": "MIT",
  "scripts": {
    "start": "node ./node_modules/pm2/bin/pm2 start ecosystem.config.js --only dev",
    "delete": "node ./node_modules/pm2/bin/pm2 delete dev",
    "logs": "node ./node_modules/pm2/bin/pm2 logs"
  },
  "dependencies": {
    "@babel/core": "^7.12.3",
    "body-parser": "latest",
    "browser-sync": "^2.26.13",
    "cors": "^2.8.5",
    "ejs": "^3.1.5",
    "express": "^4.17.1",
    "form-data": "^3.0.0",
    "html-pdf": "^2.2.0",
    "jsdoc": "^3.6.6",
    "jsonwebtoken": "^8.5.1",
    "multer": "^1.4.2",
    "node-fetch": "^2.6.1",
    "node-sass": "^5.0.0",
    "object-to-formdata": "^4.1.0",
    "pg": "^8.3.3",
```

Продолжение Приложения Е

```
"pm2": "^4.4.1",  
"qrcode": "^1.4.4",  
  "request": "^2.88.2",  
  "sequelize": "^6.3.5",  
  "winston": "^3.3.3"  
}  
}
```

Приложение Ж

Тест-кейсы

Таблица Ж.1 – Тест-кейсы для функционального тестирования

Название	Инструкция	Ожидаемый результат
Права доступа	Аутентифицироваться в панели администратора от роли «Администратор», перейти к редактированию главной страницы, нажать кнопку «Удалить»	ИС выводит ошибку о нехватке прав
Работа запросов для изменения включаемых сущностей	Аутентифицироваться в панели администратора, перейти к созданию вопроса, добавить несколько ответов на вопрос, нажать кнопку «Сохранить»	В базу данных сохранится вопрос и ответы
Пагинация	Перейти в любой раздел в панели администратора	Если записей больше 10, то отображаются кнопки пагинации. Если записей 10 или меньше, то кнопок для перехода по страницам нет
Генерация qr-кодов при создании записи	Аутентифицироваться в панели администратора, создать запись в таблице «Экспонаты»	В списке экспонатов, в каждой записи отображается ее qr-код
Генерация qr-кодов через кнопку «Сгенерировать qr-коды»	Аутентифицироваться в панели администратора, на главной странице нажать кнопку «Сгенерировать qr-коды»	Для каждой записи в таблице «Экспонаты» qr-коды сгенерируются заново
Печать qr-кодов	Аутентифицироваться в панели администратора, нажать кнопку «Распечатать», выбрать нужные qr-коды, нажать кнопку «Распечатать»	В pdf файле отобразятся выбранные qr-коды и подписи к ним
Сортировка данных	Аутентифицироваться в панели администратора, перейти в любой раздел с записями, нажать на название столбца таблицы	Данные отсортируются по выбранному столбцу
Логирование действий	Выполнить любой набор действий в панели администратора	В файл app.log сохраняются все действия произведенные в панели администратора
Просмотр файла логирования	Аутентифицироваться в панели администратора под ролью «Разработчик», нажать на кнопку «app.log»	Откроется файл app.log с информацией о действиях в панели администратора