ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИЙ

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ БИБЛИОТЕКИ ПУБЛИКАЦИЙ (НА ПРИМЕРЕ ООО «РАСТ»)

Выпускная квалификационная работа
обучающегося по направлению подготовки 38.03.05 «Бизнес-информатика»
очной формы обучения, группы 12001507
Новикова Данила Винеровича

Научный руководитель:
доц., к.э.н Ильинская Е.В.

БЕЛГОРОД 2019

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В наше время уже трудно представить организацию, не представленную в сети Интернет. Подавляющее число организаций имеют свой собственный web-сайт, который как минимум является их представительством в сети Интернет. Интернет становится неотъемлемой частью жизни современного общества. В октябре 2018 года в мире насчитывалось почти 4,2 миллиарда пользователей глобальной сети [1]. Это более половины всего мирового населения. И это число растет с каждым годом. Таким образом, в эпоху повсеместного распространения сети Интернет, когда у каждого второго человека на планете есть доступ к нему, все больше людей, прежде чем заказать услугу у той или иной компании, хотят найти эту компанию в сети Интернет.

Также сегодня люди имеют потребность делиться информацией с другими людьми. Будь то кулинарные рецепты, обзоры техники, рецензии на прочитанные книги или обширные исследования, с графиками и сводными таблицами. Однако, как правило, такой материал либо разбросан по разным узкопрофильным сайтам, либо публикуются в социальных сетях, хотя формат социальных сетей плохо подходит для написания полноценных статей, хотя бы потому что не имеют текстового редактора с достаточным функционалом форматирования текста.

Актуальность выпускной квалификационной работы заключается в том, что разрабатываемая информационная система позволит представить молодую организацию ООО «Раст» в сети Интернет, станет рекламной площадкой для ее услуг, а также ООО «Раст» расширит клиентскую базу и увеличит свой доход, за счет повышения продаж своих услуг и монетизации библиотеки публикаций. Также разрабатываемая библиотека публикаций позволит пользователям просматривать информацию об ООО «Раст», ее услугах и заказывать их, а также просматривать, оценивать, комментировать публикации и публиковать собственный материал.

Объектом исследования является процесс формирования библиотеки публикаций.

Предметом исследования являются способы хранения и рейтингования публикаций.

Целью проекта является расширение клиентской базы ООО «Раст».

Для достижения поставленной цели были сформулированы следующие задачи:

– исследовать способы хранения и рейтингования публикаций;

– исследовать теоретические аспекты проектирования и разработки web-сайта;

– провести исследование деятельности и анализ бизнес-процессов ООО «Раст»;

– исследовать и выбрать средства проектирования и разработки web-сайта;

– спроектировать и разработать web-сайт для ООО «Раст»;

– оценить экономическую эффективность разработки.

Выпускная квалификационная работа состоит из введения, трех разделов и заключения.

Введение содержит актуальность выбранной темы, объект и предмет исследования, а также цель и задачи.

В первом разделе проведено исследование теоретических аспектов проектирования и разработки web-сайта и исследованы способы хранения и рейтингования публикаций.

Во втором разделе произведено исследование деятельности и анализ бизнес процессов ООО «Раст».

В третьем разделе проведено проектирование и разработка web-сайта для ООО «Раст», а также проведена оценка экономической эффективности проекта.

В заключении подводятся итоги работы и формируются окончательные выводы.

1 Теоретические аспекты проектирования и разработки web-сайта

1.1 Исследование способов хранения и рейтингования публикаций

На сегодняшний день, во времена глобального распространения сети Интернет, люди, желающие поделиться полезной информацией с другими людьми в текстовом формате, все реже публикуют свой авторский материал в печатные издания. Сегодня такие люди, как правило, публикуют свой материал на узкопрофильных тематических сайтах, по тематике своего материала, либо публикуют свои авторский материал в социальных сетях, либо на сайтах с неограниченной тематикой и возможностью публикации.

Таким образом, можно выделить следующие способы хранения публикаций:

– в печатных изданиях;

– на узкопрофильных сайтах, с возможностью публикации;

– в социальных сетях;

– на сайтах с неограниченной тематикой и возможностью публикации.

Проанализируем каждый из способов более подробно.

1) Печатные издания.

Печатное издание – это средство массовой информации, распространяющее свою продукцию в печатном виде, то есть, отпечатанное на бумаге тем или иным способом — совокупность таких субъектов журналистской деятельности, как газеты, журналы, альманахи, сборники, бюллетени [2].

На сегодняшний день этот способ все менее привлекателен для людей, потому что процесс публикации занимает много времени и, как правило, стоит денег. К тому же, количество аудитории таких изданий постоянно уменьшается [3]. На данный момент, этот способ можно считать устаревающим.

2) Узкопрофильные web-сайты, с возможностью публикации.

Подобные web-сайты называются тематическим коллективным блогом.

Коллективный блог – это разновидность блогов, ведётся группой лиц, использующих разные учетные записи, по правилам, определяемым владельцем или модератором [4].

К таким web-сайтам, к примеру, относятся:

– Executive – интернет-издание, связанное с тематикой управления бизнесом;

– Habr – ресурс для ИТ-специалистов;

– Cossa – интернет-издание о маркетинге и коммуникациях в сфере digital.

На всех перечисленных сайтах можно публиковать собственный материал.

Однако, узкая тематика таких сайтов ограничивает авторов в публикации материала на темы, отдаленные от тематики сайта.

3) Социальные сети.

Социальная сеть – это сервис, который позволяет распространять информацию, строить социальные связи и налаживать взаимоотношения [5].

Такие web-сайты, как правило, имеют следующие недостатки:

– ограничение на количество символов публикации;

– отсутствие разделов или возможности добавления ключевых слов;

– ограниченная система рейтингования в виде оценок «нравится»;

– отсутствие текстового редактора с достаточным функционалом форматирования текста.

Таким образом, данный способ плохо подходит для публикации своего авторского материала.

4) Web-сайты с неограниченной тематикой, открытые для публикаций.

Такой сайт называется коллективным блогом с неограниченной тематикой. Этот способ отлично подходит для авторов, и не ограничивает их узкой направленностью.

Рассмотрим способы рейтингования публикаций.

На сегодняшний день, на просторах сети Интернет, распространены следующие системы оценки материала:

1) Оценка «нравится».

Такой способ преимущественно распространен в социальных сетях и имеет явный недостаток – читатель не имеет возможности оценить запись нейтрально или отрицательно, если она по каким-то критериям ему не понравилась.

2) Оценки «нравится» и «не нравится»

Такой способ имеет уже два варианта оценки – «понравилось» и «не понравилось». Что все еще недостаточно информативно для ведения статистики публикации.

3) Пятибалльная система рейтинга.

Такой способ привносит градацию оценок пользователей. Пользователь может оценить материал на «отлично» (5 баллов), «хорошо» (4 балла) и так далее. При использовании этого способа, у материала показано среднее арифметическое всех оценок, которое демонстрирует отношение большинства оценивших пользователей к конкретному материалу. На сегодняшний день этот способ оценки является наиболее объективным.

1.2 Исследование теоретических аспектов проектирования и разработки web-сайта

Так как библиотека публикаций будет представлена в виде web-сайта, необходимо рассмотреть теоретические аспекты проектирования и разработки web-сайта.

Проектирование web-сайта – это начальный и ключевой этап создания сайта. Оно экономит время и деньги. Исправить ошибку на этапе проектирования намного проще, чем сделать это на этапе разработки дизайна, вёрстки или программирования.

Только четко сформулировав задачи, определив целевую аудиторию сайта и ее потребности, смоделировав взаимодействие пользователей с сайтом, можно быть уверенным, что в результате получится именно то, что было запланировано.

Проектирование web-сайта можно условно разбить на четыре основных этапа:

– целеполагание;
– исследование контекста;
– создание концепции;
– моделирование.

Рассмотрим каждый из них подробнее.

1) Целеполагание. Данный этап необходим для определения того, зачем создается сайт и каких именно результатов необходимо достичь. Это служит ориентиром для всей дальнейшей работы [6]. В будущем они же помогут оценить успешность проекта. На этапе проектирования цели формулируются точно и подробно, а также определяются требования к разработке сайта, выполнение которых будет способствовать достижению каждой цели.

2) Исследование контекста. Этот этап необходим для получения контекстной информации сайта. Под контекстом сайта понимается различные обстоятельства, окружающие сайт и способные оказать влияние на его работу [7]. К таким обстоятельствам относятся:

– целевая аудитория и её потребности;
– характеристика и тенденции области;
– конкуренты и их деятельность;
– опыт других проектов;

– законодательные или иные ограничения;

– другие факторы влияния, в зависимости от тематики проекта.

Контекст проекта помогает определить целевую аудиторию и то, каким нужно сделать сайт: как его позиционировать, какая информация на нём должна быть и какой коммуникативной стратегии следует придерживаться.

3) Создание концепции. Концепция – это основные идеи и возможности, заложенные в проект [8]. Она задаёт направление проектированию и отвечает на вопросы:

– что и для кого создается — общая идея и целевая аудитория;

– как сайт будет работать и какую информацию содержать;

– как сайт будет зарабатывать (если это проект с прямой монетизацией);

– каковы будут отличительные особенности сайта (от конкурентов), как он будет позиционироваться;

– как сайт будет развиваться после запуска.

4) Моделирование. Моделирование web-сайта — это создание модели, которая описывает функциональные возможности и информационную структуру сайта [9].

Информационная структура — это схема, показывающая, из каких разделов состоит сайт, какие задачи они решают, и как пользователь будет перемещаться по сайту.

В функциональной части модели описываются возможности, которые сайт предоставляет своим пользователям: например, выкладывать, группировать и комментировать фотографии (социальная сеть) или заказывать и оплачивать товар (интернет-магазин).

Далее идет разработка прототипа сайта. Прототип сайта - это схематическое расположение блоков сайта, с выделением активных зон и отображением всех путей перехода [10].

Причины, по которым перед созданием сайта, желательно сначала создать его прототип:

– позволит рационально тратить время и сосредоточиваться именно на том, для чего предназначена каждая страница, а также получить четкую картину того, какая именно информация будет необходима на каждой странице сайта до разработки его дизайна;

– позволит подстраховать разработчиков, в случае неосведомленных заказчиков, которые склонны менять свое мнение на стадии разработки проекта;

– позволит избежать разногласий и недопониманий в процессе разработки сайта - клиент утверждает прототип, и значит он соглашается с тем, что должно получиться в результате;

– позволит получить четкое представление о том, как будут взаимодействовать посетители с сайтом (без цветовой схемы или элементов дизайна);

– позволит заранее выявить и удалить лишние элементы, которые могут оказаться ненужными для будущего сайта;

– прототип довольно легко создать, позволяя плавно и эффективно осуществлять процесс планирования;

– снижает вероятность увеличения объема работы по разработке дизайна;

– прототип дает дизайнеру четкое представление о том, что нужно сделать.

Созданный прототип используется при дальнейшей разработке дизайн-макета пользовательского интерфейса.

Далее начинается процесс разработки, который включает в себя следующие этапы:

– разработка дизайн-макета пользовательского интерфейса;

– техническая реализация.

Подробно рассмотрим каждый из них.

Одним из важнейших этапов создания сайта является разработка дизайн-макета пользовательского интерфейса. На этом этапе объединяются

все предыдущие стадии и web-сайт становится еще ближе к своей окончательной форме. На этом шаге дизайнер в специальной графической программе создает дизайн страниц будущего web-сайта с прорисовкой всех графических (баннеров, кнопок, фотографий) и текстовых элементов. Основное назначение дизайн-макета — визуализировать структуру страницы, ее содержимое, а также отобразить основной функционал [11]. В дизайн-макете, в отличие от прототипа, созданного на этапе проектирования, используются элементы дизайна. Дизайн-макет содержит цвета, логотипы и изображения. Он дает возможность судить о том, как в конечном результате будет выглядеть готовый сайт. Дизайнер создает дизайн web-страниц с учетом пожеланий заказчика и сформулированной информации, полученной на этапе проектирования.

После того, как дизайн-макет нарисован, согласован и утвержден, его нужно превратить непосредственно в web-сайт – перейти к технической реализации.

Традиционный процесс технической реализации состоит из следующих этапов:

– верстка;

– разработка базы данных;

– интеграция верстки и базы данных с системой управления сайтом или создание собственных программных решений.

Рассмотрим подробнее каждый из этапов:

Начальным этапом процесса технической реализации является верстка. Верстка – это перевод дизайна, до этого момента существующего в виде дизайн-макета, в HTML-код [12].

HTML - Язык разметки гипертекстовых страниц (Hypertext Markup Language) представляет собой язык, разработанный специально для создания Web-документов [13]. Он определяет синтаксис и размещение специальных инструкций (тегов), которые не выводятся на экран, но указывают браузеру, как отображать содержимое документа. Он также используется для создания

ссылок на другие документы, локальные или сетевые, например, находящиеся в сети Интернет.

Этап разработки базы данных включает в себя выбор системы управления базой данных (СУБД) и создание структуры данных, а именно: выделение сущностей, определение их свойств и связей между сущностями.

База данных – это информационная модель, позволяющая упорядоченно хранить данные о группе объектов, обладающих одинаковым набором свойств [14]. С помощью систем управления базами данных (СУБД) производится структурирование информации, находящейся в базе данных.

Система управления базами данных (СУБД) – это совокупность языковых и программных средств, с помощью которого пользователи могут создавать, определять и поддерживать базу данных, а также осуществлять к ней контролируемый доступ [15].

Сегодня средств, с помощью которых можно построить базу данных, большое множество. По структуре организации они делятся на табличные (реляционные) и иерархические. В иерархической базе данных записи упорядочиваются в определенную последовательность, как ступеньки лестницы, и поиск данных может осуществляться последовательным «спуском» со ступени на ступень. Иерархическая база данных по своей структуре соответствует структуре иерархической файловой системы. Реляционная база данных же представляет собой двумерную таблицу.

В современных СУБД используется клиент-серверный подход. СУБД представляет собой сервер, который принимает сетевые соединения, исходящие от программ-клиентов. Клиентские программы имеют возможность работать как на том же компьютере, что и программа-сервер, так и на других компьютерах [16]. Как только соединение установлено, клиент может отправлять запросы в сторону сервера, и получать от него ответы. Таким образом, СУБД отвечает только за обработку запросов и хранение данных. Всю интерактивную часть (взаимодействием с потребителем) берут на себя клиентские программы.

Рассмотрим основные функции СУБД:

– управление данными во внешней памяти (на дисках);

– управление данными в оперативной памяти с использованием дискового кэша;

– журнализация изменений, резервное копирование и восстановление базы данных после сбоев;

– поддержка языков базы данных (язык определения данных, язык манипулирования данными).

Компоненты, которые содержат современные СУБД:

– ядро - отвечает за журнализацию и управление данными во внешней и оперативной памяти;

– процессор языка базы данных – обеспечивает оптимизацию запросов на извлечение и изменение данных, а также создание машинно-независимого исполняемого внутреннего кода;

– подсистема поддержки времени исполнения - интерпретирует программы манипуляции данными, которые создают пользовательский интерфейс с СУБД;

– сервисные программы (внешние утилиты) - обеспечивают ряд дополнительных возможностей по обслуживанию информационной системы.

СУБД могут классифицироваться по типу управляемой базы данных:

– сетевые;

– объектно-реляционные;

– иерархические;

– реляционные;

– объектно-ориентированные.

По архитектуре организации хранения данных:

– локальные СУБД - все части локальной СУБД размещаются на одном компьютере;

– распределенные СУБД - части СУБД могут размещаться на двух и более компьютерах.

По способу доступа к базе данных СУБД бывают:

– файл-серверные - в файл-серверных СУБД файлы данных располагаются централизованно на файл-сервере. Ядро СУБД располагается на каждом клиентском компьютере. Доступ к данным осуществляется через локальную сеть. Синхронизация чтений и обновлений осуществляется посредством файловых блокировок. Преимуществом этой архитектуры является низкая нагрузка на ЦП сервера, а недостатком — высокая загрузка 7 локальной сети. На данный момент файл-серверные СУБД считаются устаревшими [17]. Примеры: Microsoft Access, Borland Paradox.

– клиент-серверные - состоят из клиентской части, которая входит в состав прикладной программы и сервера. В отличие от файл-серверных, они обеспечивают разграничение доступа между пользователями, мало загружают сеть и клиентские машины. Сервер является внешним программным обеспечением по отношению к клиенту, и при необходимости его можно заменить другим. Недостатки клиент-серверных СУБД в самом факте существования сервера и больших вычислительных ресурсах, потребляемых сервером [18]. Примеры: Firebird, Interbase, MS SQL Server, Oracle, MySQL;

– встраиваемые СУБД - это набор программных библиотек и компонент, предназначенных для работы с базами данных и используемых для создания приложений. Их преимуществом является тот факт, что приложения, использующие данные библиотеки, как правило, управляют локальными базами данных и независимы от серверов [19]. Примеры: SQLite, один из вариантов Firebird, один из вариантов MySQL, Sav Zigzag, Microsoft SQL Server Compact.

Следующим этапом процесса технической реализации, после выбора базы данных и создания архитектуры данных, является интеграция верстки и базы данных с системой управления сайтом или создание собственных программных решений.

Для начала нужно определить целесообразность использования системы управления сайтом или разработки собственных программных решений для решения поставленных задач. Чтобы ответить на этот вопрос, необходимо рассмотреть преимущества и недостатки каждого из подходов.

Система управления сайтом (CMS) – это программный продукт, позволяющий упростить процесс работы с сайтом, включающий в себя изменения структуры сайта, а также добавление, редактирование и удаление из него различной информации [20].

Рассмотрим основные преимущества использования CMS:

1) Доступность web-разработки.

Системы управления сайтом условно подразделяются на платные (например, 1С-Битрикс) и бесплатные, то есть CMS с открытым кодом (например, WordPress). Использовать бесплатную CMS может любой желающий.

2) Скорость создания сайтов.

Раньше создание сайта было необычайно длительным и трудоемким процессом, полным проб и ошибок. Сегодня CMS избавили вебмастеров и разработчиков от массы ненужных операций. Максимальный уровень автоматизации – максимальная скорость создания новых сайтов.

3) Простота разработки и поддержки.

Благодаря тщательно продуманному функционалу того же WordPress, сегодня не нужно быть программистом, чтобы запустить свой сайт. Удобный интерфейс CMS позволяет легко создавать шаблонные сайты, а возможность добавлять плагины позволит расширить функционал сайта.

4) Распространенность CMS.

Системы управления сайтом прочно лидируют в сайтостроительстве – подавляющее большинство web-сайтов переходит на CMS, и тенденция эта сохраняется. На сегодняшний день около 66% сайтов построены с использованием CMS [21]. Распространенность CMS автоматически

означает, что у вебмастера не возникнет проблем с техподдержкой, поиском специалистов и студий web-разработки.

5) Широкие возможности.

Фактически, системы управления сайтом превратились в конструкторы, где почти для любых задач есть свой готовый модуль.

6) Дизайн отдельно от контента.

Система поддерживает редактирование контента отдельно от функциональных элементов и дизайна. Поэтому классическая CMS позволяет технически неподготовленным пользователям добавлять и редактировать контент на web-сайте, не нарушая код.

7) Права и доступ.

CMS системы имеют гибкую настройку прав пользователей и ролей.

8) Простота настройки.

Внеся изменения в CMS, можно сразу увидеть их на всех требуемых страницах одновременно. Более того, изменения и настройки можно отслеживать по истории: кто, когда и с какой целью менял те или иные параметры.

9) Регулярные обновления.

Платные CMS обеспечивают вам непрерывный доступ к обновлениям и полноценную техническую поддержку web-сайта. К сожалению, это не относится к системам с открытым кодом – они весьма уязвимы и не получают регулярных обновлений.

Несмотря на множество достоинств, системы управления сайтом имеют серьезные недостатки. Рассмотри их подробнее:

1) Ограниченность.

Несмотря на обширный выбор расширений и плагинов, некоторые бизнес-задачи требуют индивидуальной разработки того или иного модуля под заказ. С этой точки зрения CMS существенно ограничивает вебмастера в творческой свободе.

2) Низкая безопасность сайта.

Распространенность CMS во всем мире – это одновременно успех и уязвимость. Внутреннее устройство наиболее популярных систем изучено хакерами вдоль и поперек, что делает ваш сайт потенциальной жертвой очередного злоумышленника. Конечно, ведущие компании-разработчики активно борются за безопасность и применяют контрмеры. Здесь и проявляется важное отличие между платными и бесплатными CMS.

3) Однотипность сайтов.

Сделать web-сайт на CMS в точности таким, каким вы представляете в мечтах, будет довольно сложно. Да, разные системы имеют разную степень гибкости, но абсолютно каждой присуща некоторая «шаблонность».

4) Плохая семантика HTML кода.

HTML код, сгенерированной с помощью CMS, часто семантически неверен. Например, родительский блок статьи может быть задан обычным « », вместо семантически более правильного «<article>». Сайты с семантически неверным кодом хуже ранжируются поисковыми системами.

5) Медленная загрузка.

CMS хранит все ресурсы отдельно, сопоставляя их «на лету» при обращении web-клиента к определенной странице. Это означает медленную загрузку.

Рассмотрим другой подход – разработка собственного серверного «движка» для сайта. Основные преимущества данного подхода следующие:

1) Отсутствие ограничений.

Практическое отсутствие ограничений в выборе структуры, дизайна и функциональности. Отсутствие привязки к CMS дает разработчикам возможность реализовать любые задумки.

2) Нетребовательность к ресурсам.

Сайты, качественно написанные «вручную», быстро загружаются и стабильно работают на пиках посещаемости.

3) Отсутствие необходимости платить за CMS и расширения.

Однако, недостатки такого подхода очевидны:

1) Необходимы знания.

Создание сайта с нуля подразумевает не только наличие знаний языков программирования, но и умения правильно спроектировать его внутреннюю архитектуру.

2) Необходимо время.

Создание хорошего качественного сайта - это трудоемкая работа, и на ее выполнение уходит много времени. Кроме того, что необходимо построить прототип сайта, проработать дизайн, внутреннюю архитектуру, в конце концов его нужно еще и запрограммировать.

3) Необходимы деньги.

Не шаблонный сайт, качественно разработанный вручную с нуля – стоит дорого.

Таким образом, в данном разделе были исследованы способы хранения и рейтингования публикаций. В результате был выбран способ хранения публикаций в виде web-сайта с неограниченной тематикой публикаций и пятибалльная система рейтингования публикаций. Также были исследованы теоретические аспекты проектирования и разработки web-сайтов и получена теоретическая информация для создания будущего web-сайта для ООО «Раст».

2 Анализ деятельности ООО «Раст»

2.1 Исследование деятельности ООО «Раст»

ООО «Раст» – это молодая организация, зарегистрированная 18 мая 2018 года, занимающаяся работой с текстом, а именно:

– Написанием уникального текста;

– Редактированием текста;

– Профессиональным переводом текста.

Организация сотрудничает с частными и корпоративными клиентами. Среднемесячный доход ООО «Раст» составляет 300000 рублей. Юридический адрес: Россия, Белгородская область, г. Белгород, бульвар Народный, дом 111, офис 406А, индекс – 308015.

Предприятие ООО «Раст» имеет одного учредителя, который является генеральным директором организации. Предприятие имеет организационно-правовую форму – общество с ограниченной ответственностью. Деятельность предприятия осуществляется в соответствии с требованиями и ограничениями действующего законодательства РФ в области регулирования договорных отношений, налогообложения и бухгалтерского учета, трудовых отношений. Основным внутренним документов, регулирующим деятельность предприятия является его устав. Предпринимательская деятельность организации подтверждается свидетельством о внесении организации в ЕГРЮЛ.

ЕГРЮЛ (Единый государственный реестр юридических лиц) – это государственный реестр Российской Федерации, содержащий сведения обо всех юридических лицах, зарегистрированных на территории Российской Федерации, а также сведения о внесении изменений в учредительные документы юридических лиц или об их ликвидации [22].

Предприятие ведет свою деятельность по ОКВЭД.

ОКВЭД – общероссийский классификатор видов экономической деятельности, представляющий собой совокупность различных направлений деятельности, которые могут осуществлять юридические лица и индивидуальные предприниматели, причем каждое направление имеет свой код [23].

Основным видом деятельности ООО «Раст» является деятельность по обработке данных, предоставление услуг по размещению информации и связанная с этим деятельность.

Дополнительные виды деятельности ООО «Раст»:

– 46.18 деятельность агентов, специализирующихся на оптовой торговле прочими отдельными видами товаров;

– 62.09 деятельность, связанная с использованием вычислительной техники и информационных технологий, прочая;

– 63.11.1 деятельность по созданию и использованию баз данных и информационных ресурсов;

– 63.12 деятельность web-порталов;

– 63.91 деятельность информационных агентств;

– 69.10 деятельность в области права;

– 69.20 деятельность по оказанию услуг в области бухгалтерского учета, по проведению финансового аудита, по налоговому консультированию;

– 70.22 консультирование по вопросам коммерческой деятельности и управления;

– 73.11 деятельность рекламных агентств;

– 73.20 исследование конъюнктуры рынка и изучение общественного мнения;

– 74.30 Деятельность по письменному и устному переводу;

– 82.99 деятельность по предоставлению прочих вспомогательных услуг для бизнеса, не включенная в другие группировки.

Основной целью деятельности организации является получение прибыли, как основного показателя деятельности предприятия.

Учет результатов экономической деятельности ведется согласно РСБУ.

Право осуществлять правовую деятельность от лица организации принадлежит генеральному директору компании согласно ее уставу.

Прибыль от деятельности компании идет на поддержание экономических процессов предприятия и покрытие постоянных затрат. Способы использования чистой прибыли предприятия определяются генеральным директором на свое усмотрение.

На данный момент штат организации состоит из 12 человек. Организационная структура ООО «Раст» включает 4 отдела, имеющих собственные структурные единицы, а также руководителя – генерального директора.

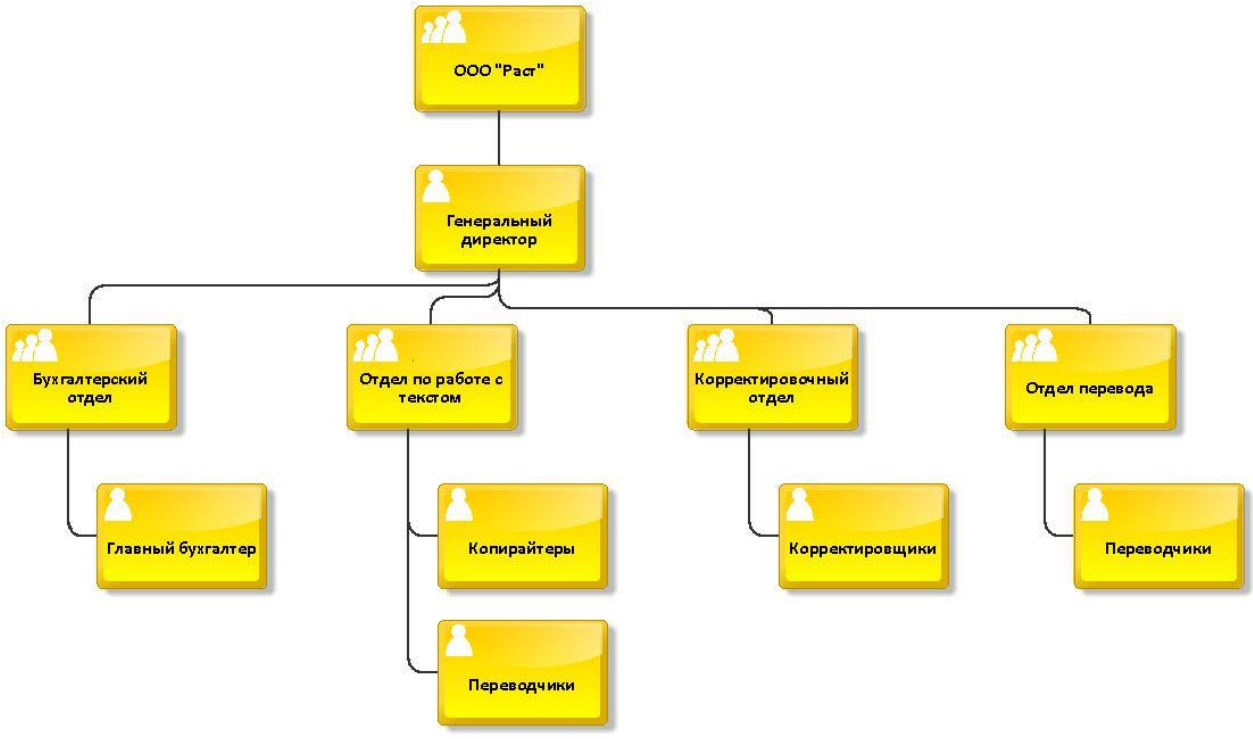

Рисунок 2.1 - Организационная структура ООО «Раст»

Каждое из структурных единиц выполняет свои должностные обязанности:

1) Генеральный директор:

– занимается ведением внутреннего и внешнего документооборота;

– определяет размер штаба подчиненных и их заработной платы, а также авансов, премий и прочих выплат;

– заключает договора с физическими и юридическими лицами;

– принимает решение о принятии и увольнении сотрудников;

– представляет правовые и экономические интересы компании;

– управляет деятельность компании в целом.

2) Бухгалтерский отдел – главный бухгалтер:

– ведет бухгалтерскую отчетность;

– занимается выплатами сотрудникам;

– принимает к учету средства, полученные в ходе экономической деятельности предприятия;

– занимается налоговыми и пенсионными платежами;

– производит расчет с контрагентами.

3) Отдел по работе с текстом – копирайтеры:

– занимаются созданием текстов;

4) Отдел по работе с текстом – корректировщики:

– отвечают за качество выполнение работ сотрудниками отдела;

– отвечают за выявление и исправление ошибок в текстах;

5) Отдел по работе с текстом – переводчики

– занимаются переводом текста.

2.2 Анализ бизнес-процессов ООО «Раст»

На данный момент основным видом деятельности ООО «Раст» являются услуги по работе с текстом (написание, редактирование и перевод текста), в которых можно проследить недостатки и предложить мероприятия по устранению этих недостатков.

Для описания первого бизнес-процесса была выбрана методология BPMN.

BPMN (англ. Business Process Model and Notation, нотация и модель бизнес-процессов) — это система условных обозначений (нотация) и их описания в XML для моделирования бизнес-процессов.

Основной целью BPMN является обеспечение доступной нотацией описания бизнес-процессов всех пользователей: от аналитиков, создающих схемы процессов, и разработчиков, ответственных за внедрение технологий выполнения бизнес-процессов, до руководителей и обычных пользователей, управляющих этими бизнес-процессами и отслеживающих их выполнение. Таким образом, BPMN нацелен на устранение расхождения между моделями бизнес-процессов и их реализацией.

Рассмотрим первый основной бизнес-процесс ООО «Раст» - «поиск и консультирование клиента» (рисунок 2.2).

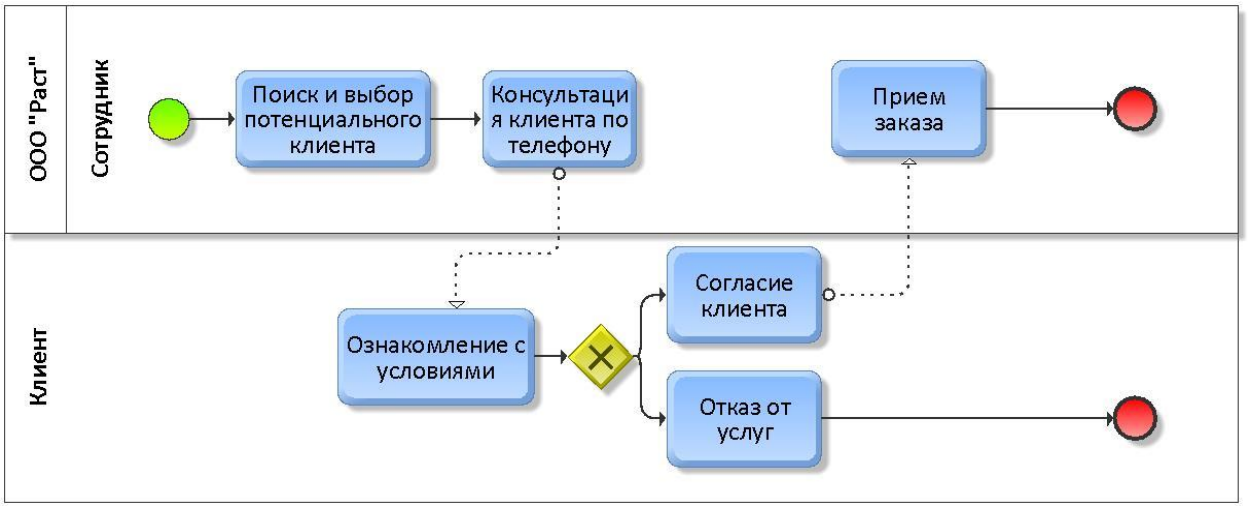

Рисунок 2.2 – Процесс поиска и консультирование клиента (BPMN)

В этот процесс входят 2 участника: сотрудник ООО «Раст» и потенциальный клиент.

Сначала сотрудник производит поиск и выбор потенциального клиента на таких сайтах как Avito, Юла, HeadHunter.

Далее сотрудник ООО «Раст» совершает телефонный звонок потенциальному клиенту.

Далее происходит процесс ознакомления клиента с предоставляемыми услугами, их ценами и сроками выполнения работ.

В случае, если клиент не согласен с условиями, то процесс завершается.

Если клиент соглашается с условиями, то сотрудник вносит данные о заказе в программу Microsoft Excel. После чего процесс завершается.

В среднем на процесс поиска и информирования одного клиента работник ООО «Раст» тратит 25 минут рабочего времени. Сюда входит время на поиск клиента (15 минут) и консультацию (10 минут).

Второй бизнес-процесс будет представлен с использованием методологии IDEF0

IDEF0 – это методология функционального моделирования и графическая нотация, предназначенная для формализации и описания бизнес-процессов.

Рассмотрим второй основной бизнес-процесс ООО «Раст» - «написание статьи».

На рисунке 2.3. представленная контекстная диаграмма IDEFO процесса написания статьи.

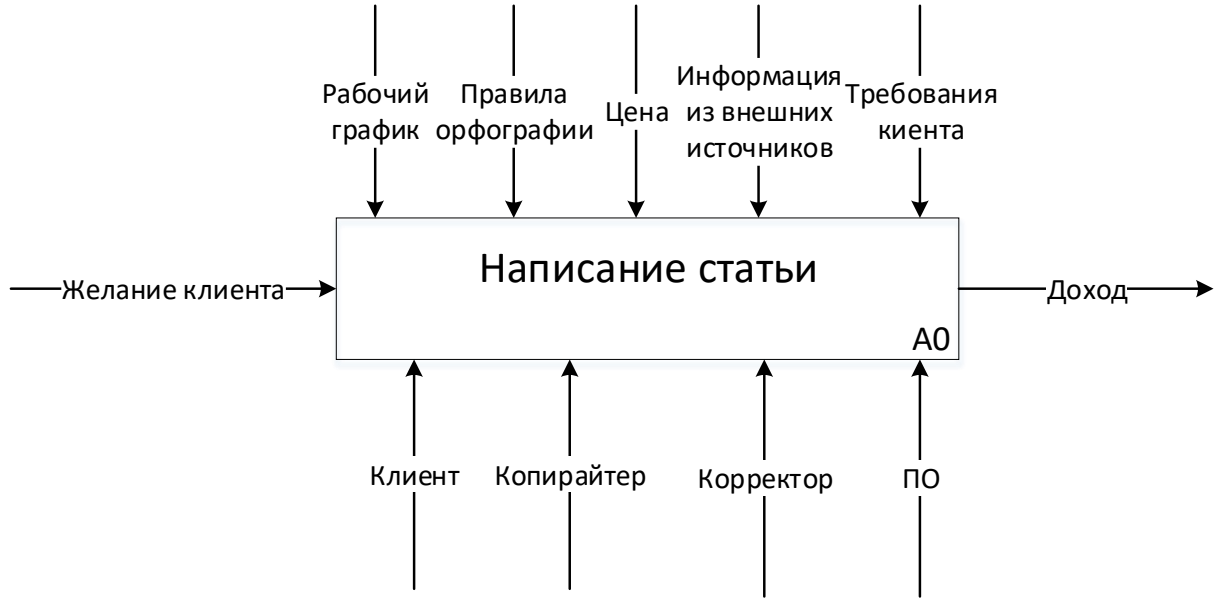

Рисунок 2.3 - Контекстная диаграмма процесса «Написание статьи» (IDEF0)

Входными данными процесса написания статьи является желание заказчика. Выходными данными является доход. Управляющим воздействием является:

- рабочий график;
- цена;
- правила орфографии;
- информация из внешних источников;
- требования заказчика;

Механизмом является:

- копирайтер;
- корректор;
- программное обеспечение;
- заказчик.

Процесс, рассмотренный выше, можно детализировать. Декомпозиция данного процесса представлена на рисунке 2.4.

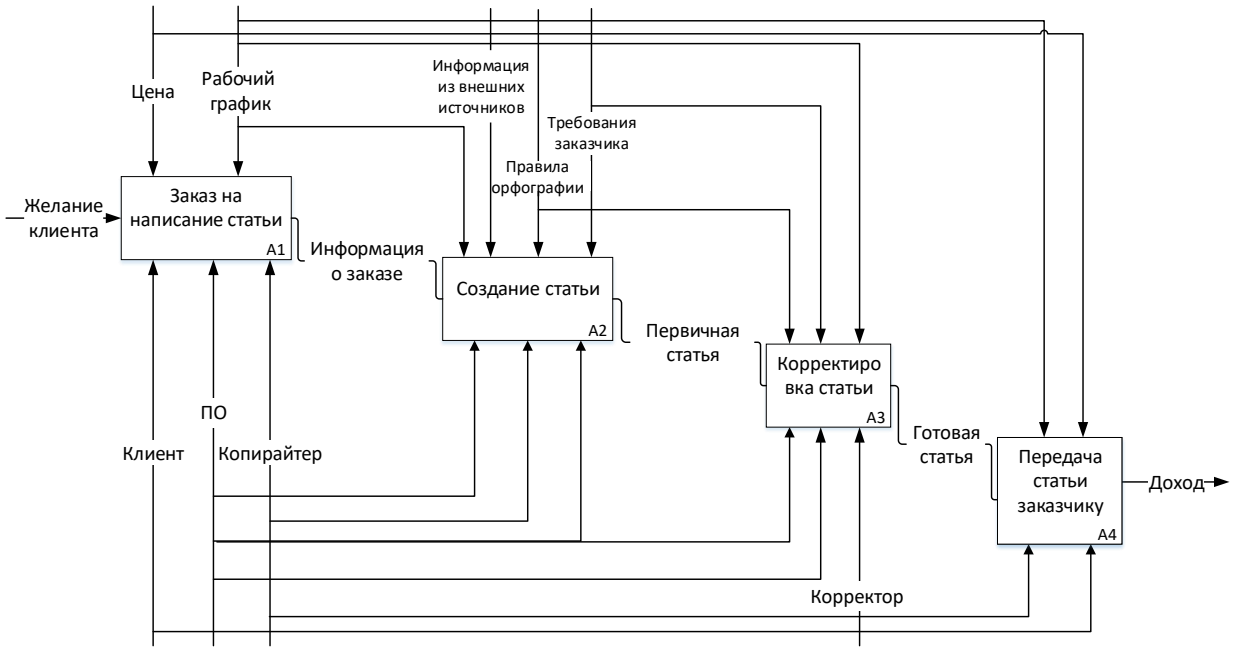

Рисунок 2.4 – Декомпозиция процесса написания и продажи статьи (IDEF0)

Рассмотрим подпроцесс «заказ на написание статьи». Входными данными этого процесса является желание клиента. Выходными данными является информация о заказе.

Управляющим воздействием является:

– рабочий график;

– цена.

Механизмами являются:

– заказчик;

– программное обеспечение;

– свободный сотрудник.

Рассмотрим подпроцесс «создание статьи». Данными на вход являются выходные данные из предыдущего процесса, то есть информация о заказе. Выходными данными является первичная статья.

Управляющим воздействием является:

– рабочий график;

– информация из внешних источников

– правила орфографии

– требования заказчика.

Механизмами являются:

– программное обеспечение;

– копирайтер.

Рассмотрим следующий подпроцесс – «Корректировка текста». Входные данные - первичная статья. На выходе получаем готовую статью.

Управляющим воздействием является:

– рабочий график;

– информация из внешних источников;

– правила орфографии;

– требования заказчика.

Механизмами являются:

– программное обеспечение;

– корректор.

Рассмотрим последний подпроцесс – «передача статьи заказчику». Входные данные - готовая статья. На выходе получаем доход.

Управляющим воздействием является:

– рабочий график;

– цена.

Механизмами являются:

– свободный сотрудник;

– заказчик.

Таким образом, в данном разделе была рассмотрена организация ООО «Раст», ее деятельность и организационная структура. В ходе чего была получена необходимая информация для информационного наполнения сайта сведениями об организации. После проведения анализа бизнес-процессов ООО «Раст», было обнаружено, что процесс поиска и информирования клиента, который занимает значительную часть рабочего времени, можно оптимизировать с помощью разрабатываемого сайта.

3 Реализация web-сайта

3.1 Проектирование web-сайта

Целью проектирования является –web-сайт для ООО «Раст», который представит организацию в сети Интернет, станет рекламной площадкой для ее услуг, а также позволит ООО «Раст» расширить клиентскую базу и увеличить свой доход, за счет повышения продаж своих услуг и монетизации библиотеки публикаций. Разрабатываемый сайт позволит пользователям просматривать информацию об ООО «Раст», ее услугах и заказывать их, а также просматривать, оценивать, комментировать публикации и публиковать собственный материал в разделе «публикации».

Обозначим требования к разработке:

1) Реализовать раздел «О нас», на которой будет представлена информация об ООО «Раст»;

2) Реализовать раздел «Услуги», в котором будет представлен перечень услуг ООО «Раст», их цены и возможность оставления заявки на обратный звонок;

3) Реализовать регистрацию и авторизацию пользователей;

4) Проработать группы прав пользователей;

5) Разработать кабинет управления заказами для администратора;

6) Предусмотреть личный кабинет пользователя, с возможностью изменения личной информации;

7) Реализовать систему публикации записей;

8) Реализовать систему комментирования публикаций для авторизованных пользователей;

9) Реализовать систему рейтингования публикаций для авторизованных пользователей.

10) Web-сайт должен быть разработан без использования CMS систем;

Разрабатываемый web-сайт позволит оптимизировать бизнес-процесс поиска и консультирования клиента, рассмотренный в разделе 2.2. Бизнес-процесс поиска и консультирования клиента с использованием разработанного web-сайта представлен на рисунке 3.1.

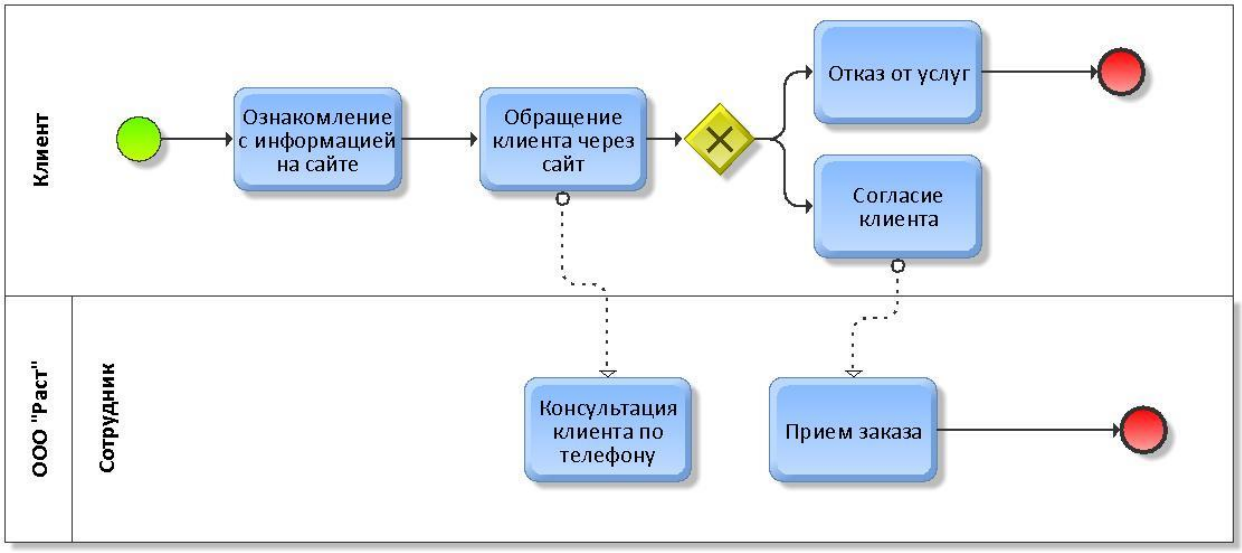

Рисунок 3.1 - Процесс поиска и консультирования клиента с использованием разработанного web-сайта (BPMN)

Благодаря разрабатываемому web-сайту и дальнейшему планируемому проведению рекламной компании, освободится рабочее время сотрудников, осуществляемое ранее на поиск и консультацию потенциального клиента по телефону. Заинтересованный клиент, с помощью web-сайта сможет самостоятельно ознакомится с услугами, предоставляемыми ООО «Раст» и их стоимостью. Далее клиент связывается с одним из сотрудников по номеру телефона, и консультация 1 клиента, первично ознакомленного с необходимой информацией, сократится и потребует у работника в среднем 5 минут рабочего времени.

Сайт рассчитан на русскоговорящую аудиторию, так как он будет представлен исключительно на русском языке. Однако, целевая аудитория разрабатываемого проекта может быть разнообразна.

Рассмотрим законодательные и иные ограничения, действующие на деятельность разрабатываемого сайта:

– 149-ФЗ «Об информации, информационных технологиях и о защите информации»;

– 152-ФЗ «О защите персональных данных»;

– 38-ФЗ «О рекламе».

Концепция разрабатываемого проекта сформулирована следующим образом – это web-сайт, который представит ООО «Раст» и ее услуги в сети Интернет и позволит пользователям заказывать их, а также в разделе позволит зарегистрированным пользователям просматривать, оценивать и комментировать публикации, а также публиковать собственный материал в разделе «публикации».

На рисунке 3.2 показана структура разделов разрабатываемого web-сайта.

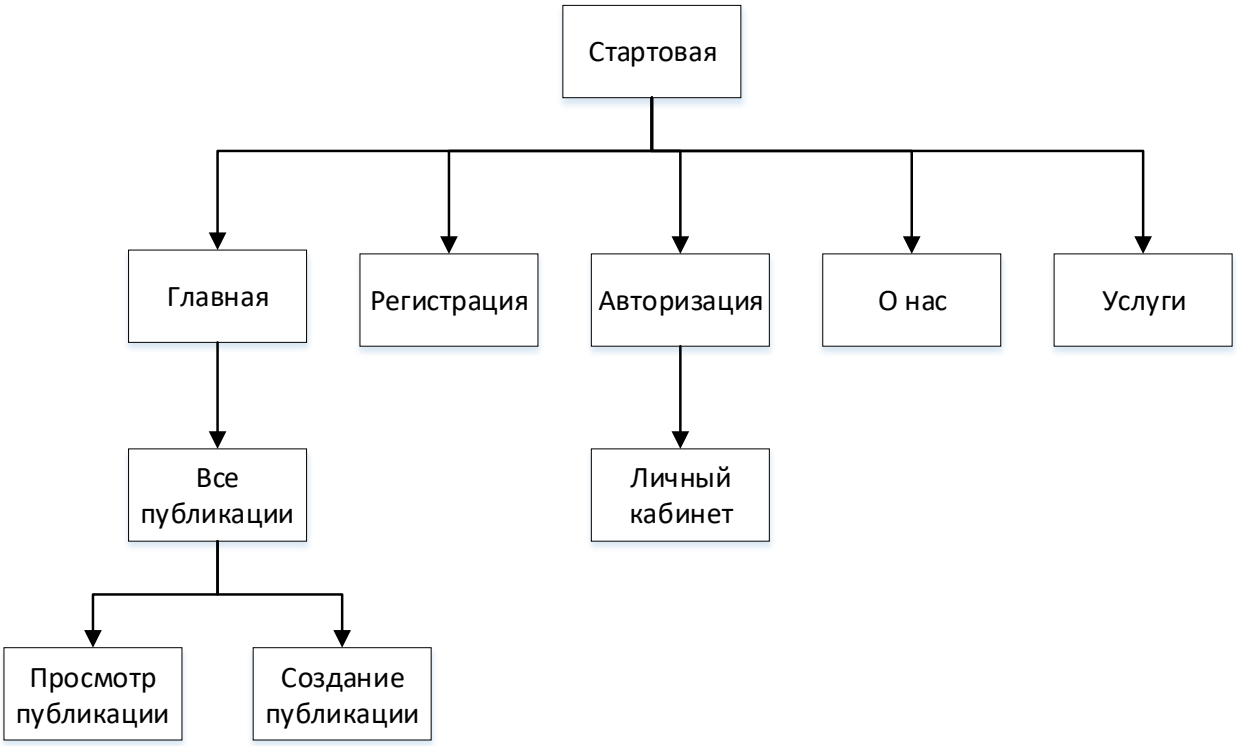

Рисунок 3.2 – Структура разделов web-сайта

Детальное описание разделов, показанных на рисунке 3.2, будет произведено при создании прототипов страниц.

На данном этапе необходимо создать прототип сайта. Нарисовать прототип можно и на бумаге, однако это может занять больше времени, нежели использование программных средств, специально предназначенных для этого. Такие программы позволяют быстро и наглядно представить будущий сайт в виде прототипа. К тому же, такие программы позволяют документировать различные состояния одного экрана, включая анимацию, переходы и мелкие взаимодействия, которые трудно отобразить в статической документации.

Далее будут рассмотрены популярные программные средства для прототипирования web-сайтов: Axure, Figma, Proto.io (таблица 3.1).

Таблица 3.1 – Сравнительный анализ программных средств для прототипирования сайтов

Критерий	**Axure**	**Figma**	**Proto.io**
Бесплатный тариф	Пробный период – 30 дней	Да	Пробный период – 15 дней
Самый дешевый тариф	29$/месяц	12$/месяц	24$/месяц
Особенности	Одно из самых известных приложений для создания прототипов. Имеется возможность запрограммировать поведение кнопок, контейнеров, виджетов.	Облачный сервис, который может использоваться как для создания прототипов, так и для создания дизайн-макета.	Возможность просматривать прототипы в офлайн-режиме с помощью мобильных приложений.
Интеграция с другими приложениями	Нет	Zeplin	Photoshop, Sketch, Dropbox
Приложение	MacOS, Windows	MacOS, Windows	Нет
Web-версия	Нет	Да	Да
Мобильное приложение	Нет	Да. Android и iOS	Да. Android и iOS

В результате проведения сравнительного анализа программных средств для прототипирования сайтов был выбран сервис Figma. Он является условно

бесплатным, что позволяет работать над одним проектом одновременно двум дизайнерам или проектировщикам. При добавлении дополнительных пользователей к проекту или при необходимости дополнительных возможностей, таких как библиотека компонентов или неограниченная история изменений, придется платить 12 долларов в месяц, что в таком случае меньше, чем у конкурентов, участвующих в сравнительном анализе. Сервис подходит как для создания интерактивных прототипов, так и для создания дизайн-макетов пользовательского интерфейса, что является несомненным преимуществом данного сервиса.

Приступим к созданию прототипов пользовательского интерфейса web-сайта, с помощью выбранного сервиса Figma.

1) Стартовый раздел (рисунок 3.3). Предназначен для ознакомления посетителей с концепцией сайта. На заднем плане страницы должно воспроизводиться видео рекламного характера. Из этой страницы можно совершить переход на главную страницу, страницу авторизации и регистрации, страницу «о нас», страницу услуг и всех публикаций.

Рисунок 3.3 – Прототип стартового раздела

2) Модальное окно регистрации (рисунок 3.4). Предназначено для регистрации пользователей на сайте. Имеет кнопку переключения на модальное окно авторизации, а также кнопку закрытия окна.

Модальное окно – это окно, которое блокирует работу пользователя с родительским приложением до тех пор, пока пользователь это окно не закроет [24].

Рисунок 3.4 – Прототип модального окна регистрации

3) Модальное окно авторизации (рисунок 3.5). Предназначено для авторизации зарегистрированных пользователей в системе. Имеет кнопку переключения на модальное окно регистрации, а также кнопку закрытия окна.

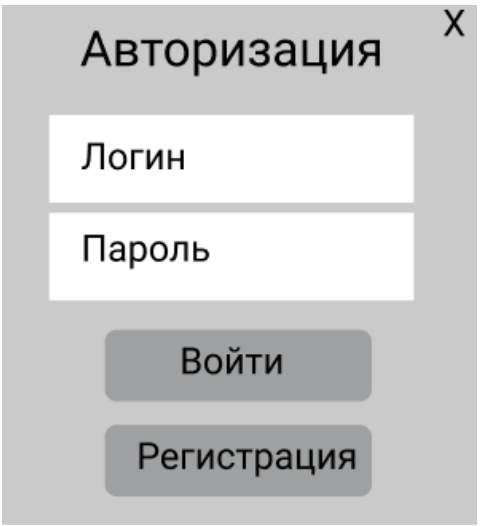

Рисунок 3.5 – Прототип модального окна авторизации

4) Главная страница (рисунок 3.6). Размещает слайдер с фотографиями, статистику, описание сайта и ссылки на официальные

страницы в социальных сетях. На этой странице (и на всех последующих) присутствует боковое навигационное меню, которое имеет 2 положения – «открытое» и «закрытое».

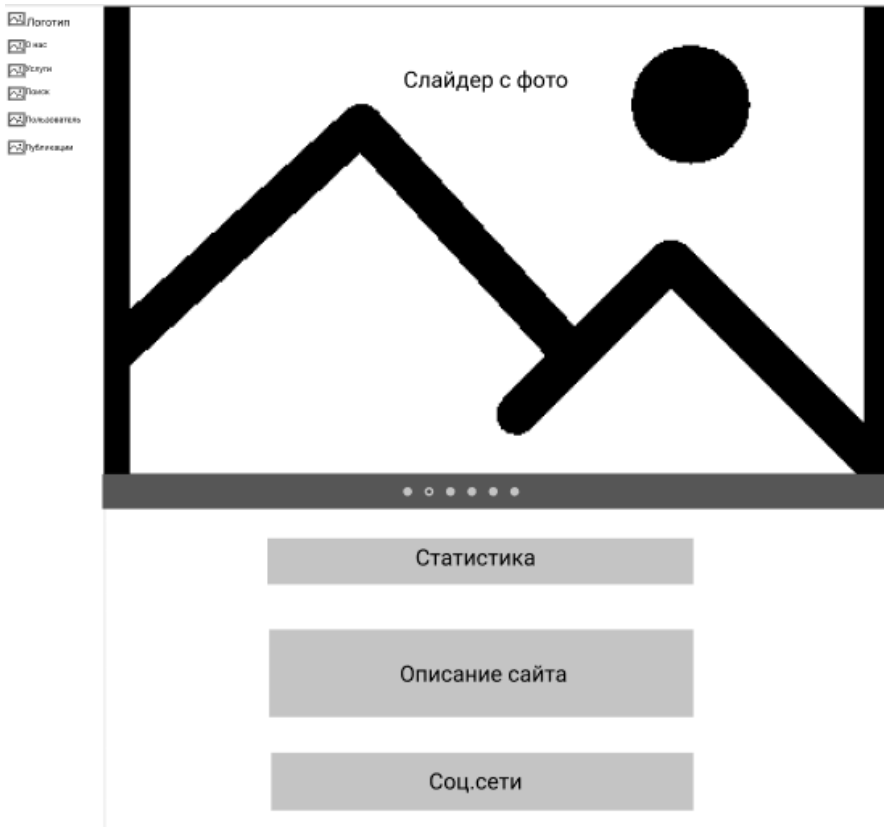

Рисунок 3.6 – Прототип главной страницы

5) Боковое навигационное меню в открытом положении (рисунок 3.7). Предназначено для быстрого перехода пользователя в интересующий раздел. С помощью этого меню можно совершить переход в главный раздел (по нажатию на логотип), раздел «о нас», раздел услуг, личный кабинет пользователя и в раздел публикаций.

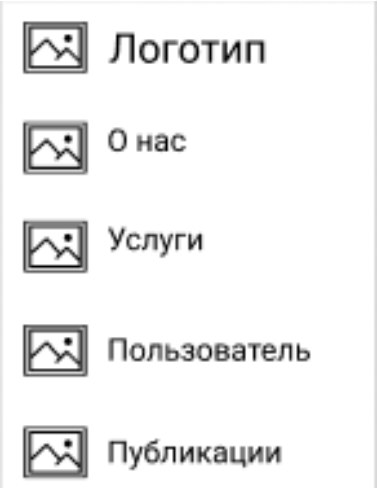

Рисунок 3.7 – Прототип бокового меню

6) Раздел «О нас» (рисунок 3.8). Предназначен для ознакомления пользователя с информацией об ООО «Раст». Содержит кнопки перехода на страницу услуг и ссылки на социальные сети организации.

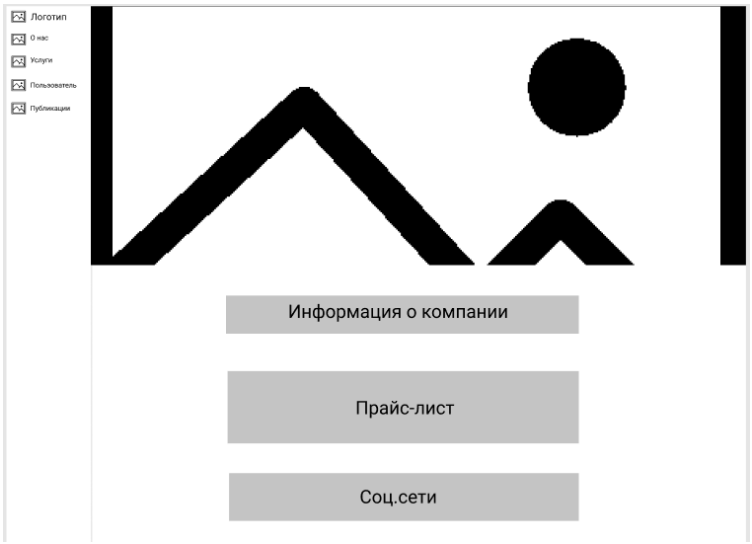

Рисунок 3.8 – Прототип раздела «О нас»

7) Раздел услуг (рисунок 3.9). Предназначен для ознакомления со списком услуг, предоставляемыми ООО «Раст» и их ценами. Имеет кнопку «связь», с помощью которой можно оставить заявку на звонок.

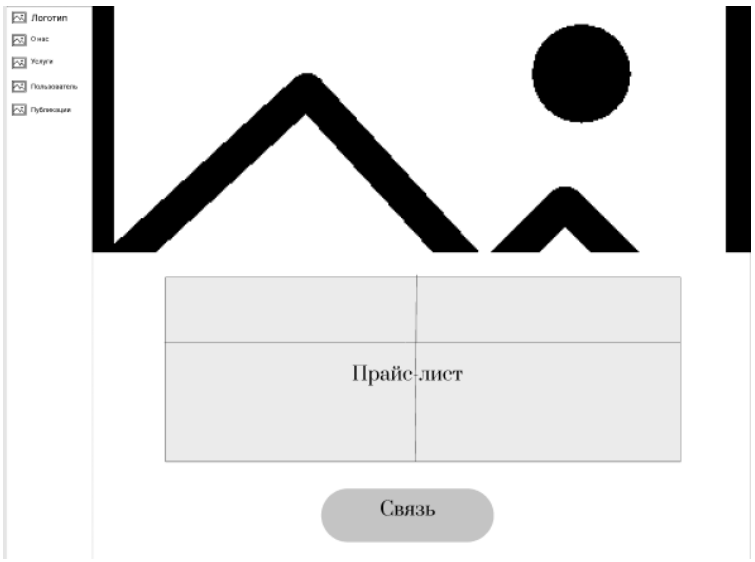

Рисунок 3.9 – Прототип раздела услуг

8) Раздел публикаций (рисунок 3.10). Содержит все опубликованные пользователями материалы. На этой странице присутствует верхнее меню, которое позволяет применять сортировку и фильтры, а авторизованному пользователю написать публикацию.

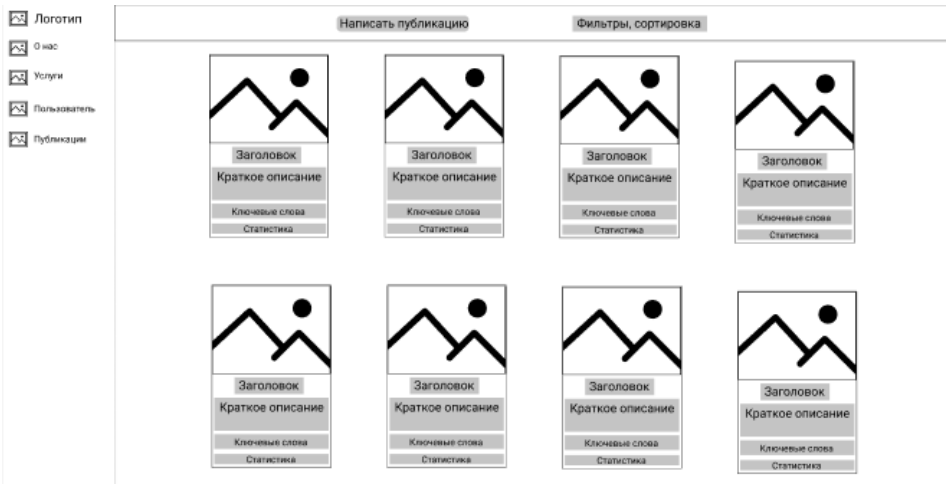

Рисунок 3.10 – Прототип раздела публикаций

9) Раздел просмотра конкретной публикации (рисунок 3.11). Позволяет просматривать конкретную публикацию в полном объеме, а также оценить ее и прокомментировать. В данном разделе отображена статистка публикации, в частности пятибалльный рейтинг, вычисляющийся как среднее арифметическое всех оценок и их количества.

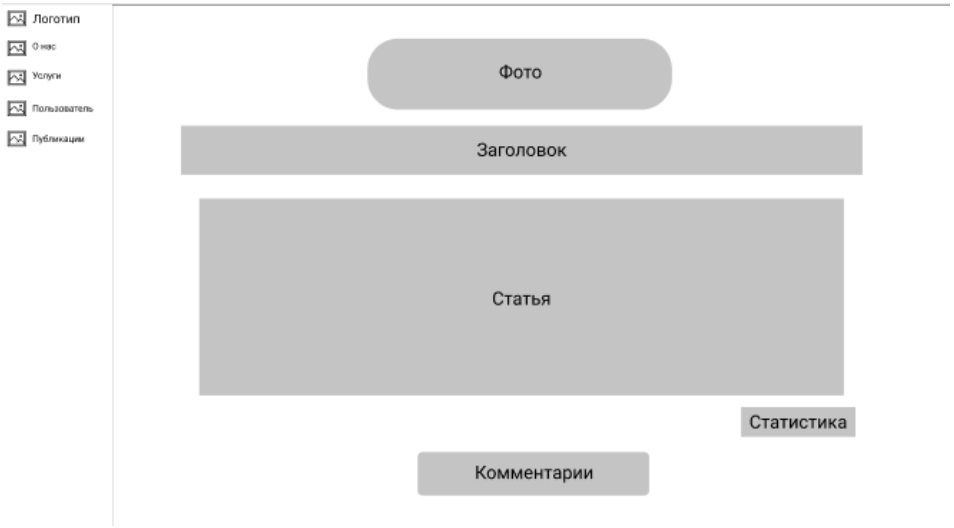

Рисунок 3.11 – Прототип раздела просмотра конкретной публикации

10) Раздел создания публикации (рисунок 3.12). Позволяет создавать публикацию. На странице расположен текстовый редактор с широким функционалом для форматирования текста.

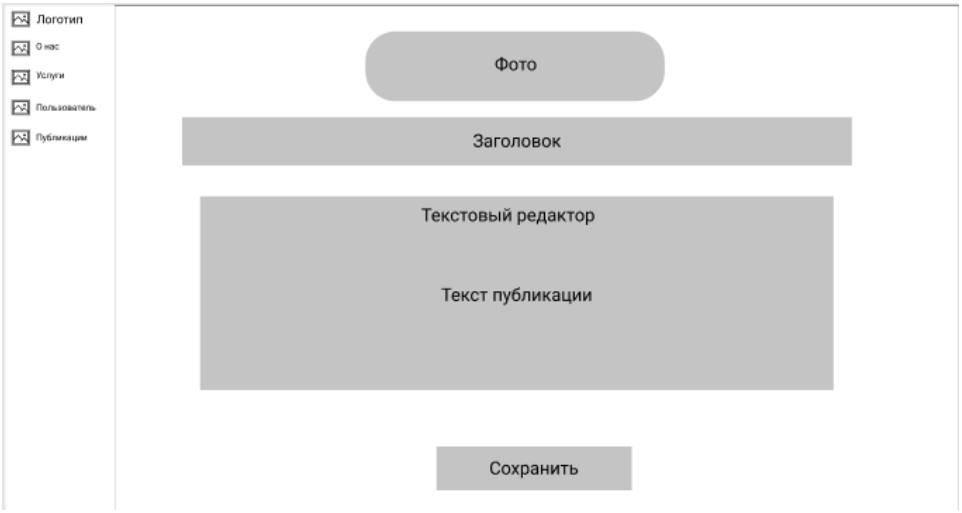

Рисунок 3.12 – Прототип раздела создания публикации

11) Личный кабинет пользователя (рисунок 3.13). Позволяет зарегистрированным пользователям просматривать и изменять личные данные, а также видеть статистику своих публикаций.

Рисунок 3.13 – Прототип личного кабинета пользователя

12) Кабинет управления заказами (рисунок 3.14). Позволяет администраторам просматривать заказы и помечать их выполнение.

Рисунок 3.14 – Прототип кабинета управления заказами

Далее была создана диаграмма прецедентов пользователей, которая представлена на рисунке 3.15.

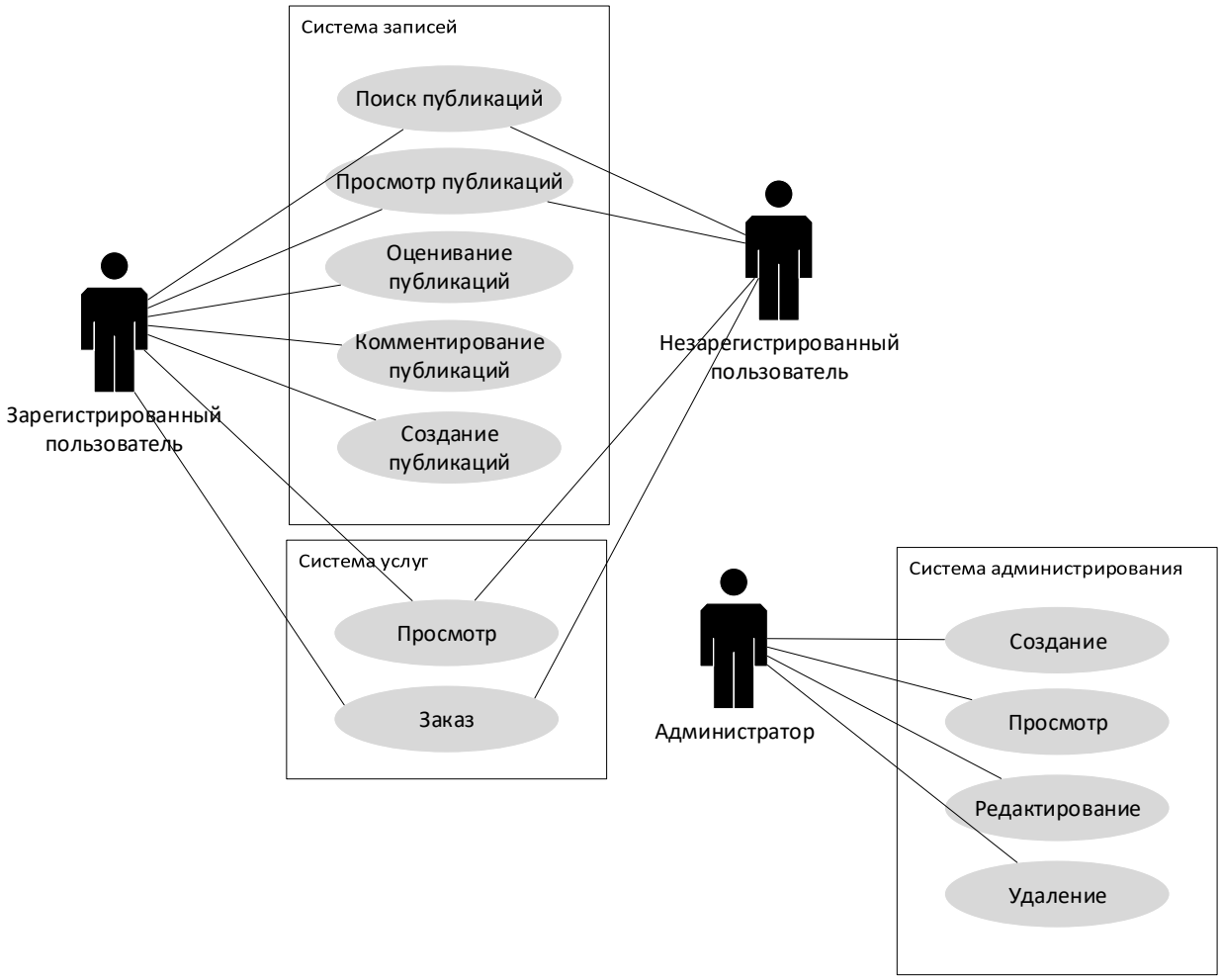

Рисунок 3.15 – Варианты поведения пользователей

В системе записей незарегистрированный пользователь может осуществлять поиск публикаций, а также просматривать их.

В системе записей зарегистрированный пользователь может осуществлять поиск публикаций, просмотр, оценивание, комментирование, а также создание публикации.

В системе администрирования администратор может создавать, просматривать, редактировать и удалять любые объекты сайта.

В системе услуг пользователи могут просматривать и совершать заказ услуг.

3.2 Разработка web-сайта

Первый этап разработки web-сайта – создание дизайн-макетов пользовательского интерфейса. Для создания web-дизайна, дизайнеру, как и художнику, без качественных инструментов труда работать очень сложно. Если художник, к примеру, использует кисти и краски, то создатель сайтов – специальные программы.

Выбранный ранее инструмент для прототипирования Figma, имеет богатый функционал и подходит для создания дизайн-макетов пользовательского интерфейса, поэтому он использовался также и для создания дизайна web-сайта для ООО «Раст».

1) Стартовый раздел (рисунок 3.16).

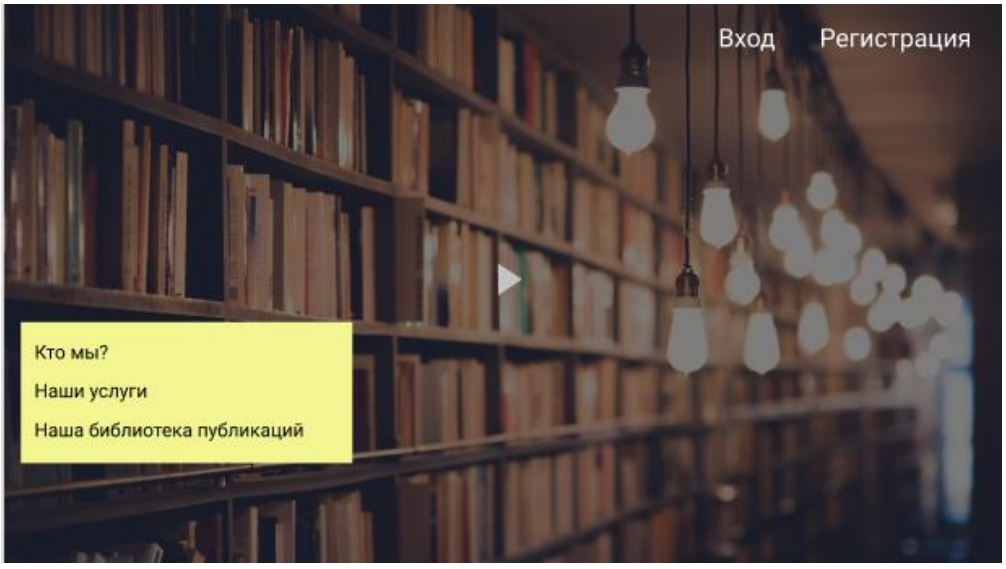

Рисунок 3.16 – Дизайн стартового раздела

2) Модальное окно авторизации (рисунок 3.17).

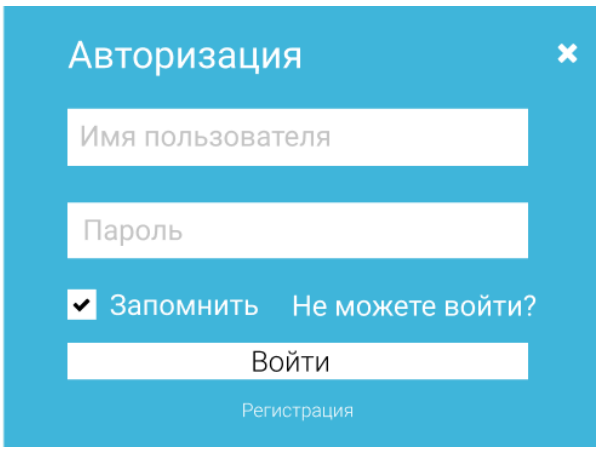

Рисунок 3.17 – Дизайн модального окна авторизации

3) Модальное окно авторизации (рисунок 3.18).

Рисунок 3.18 – Дизайн модального окна регистрации

4) Дизайн главного раздела (рисунок 3.19). На этой и последующих страницах имеется боковое навигационное меню.

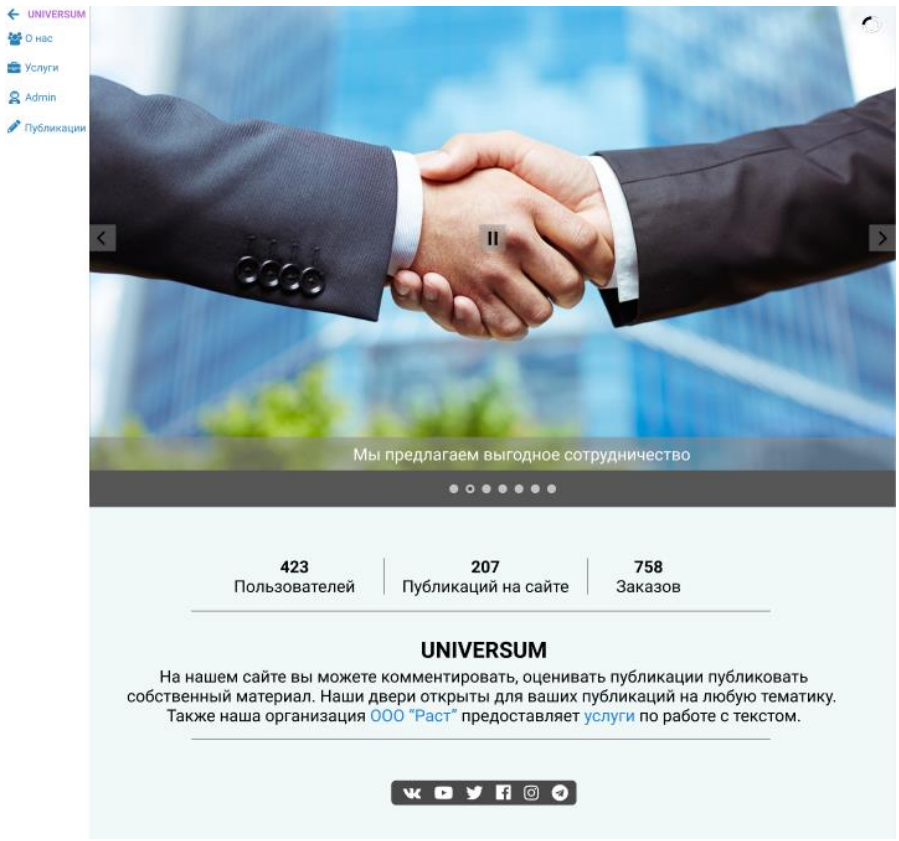

Рисунок 3.19 – Дизайн главного раздела

5) Дизайн бокового навигационное меню в открытом положении (рисунок 3.20).

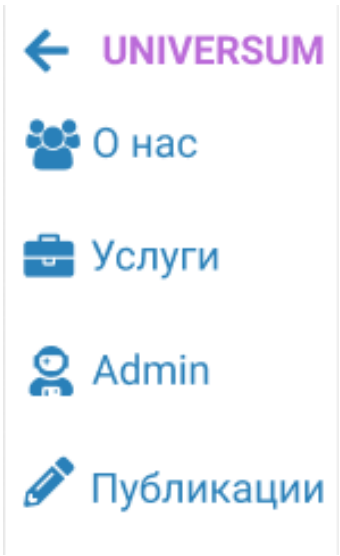

Рисунок 3.20 – Дизайн бокового навигационного меню в открытом положении

6) Дизайн раздела «О нас» (рисунок 3.21).

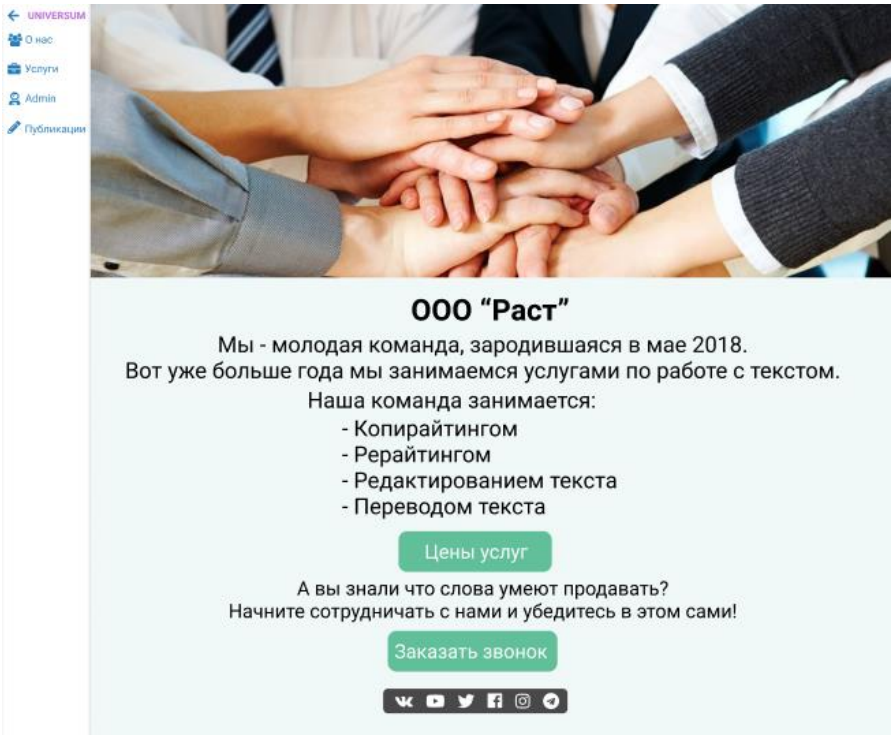

Рисунок 3.21 – Дизайн раздела «О нас»

7) Дизайн раздела услуг (рисунок 3.22).

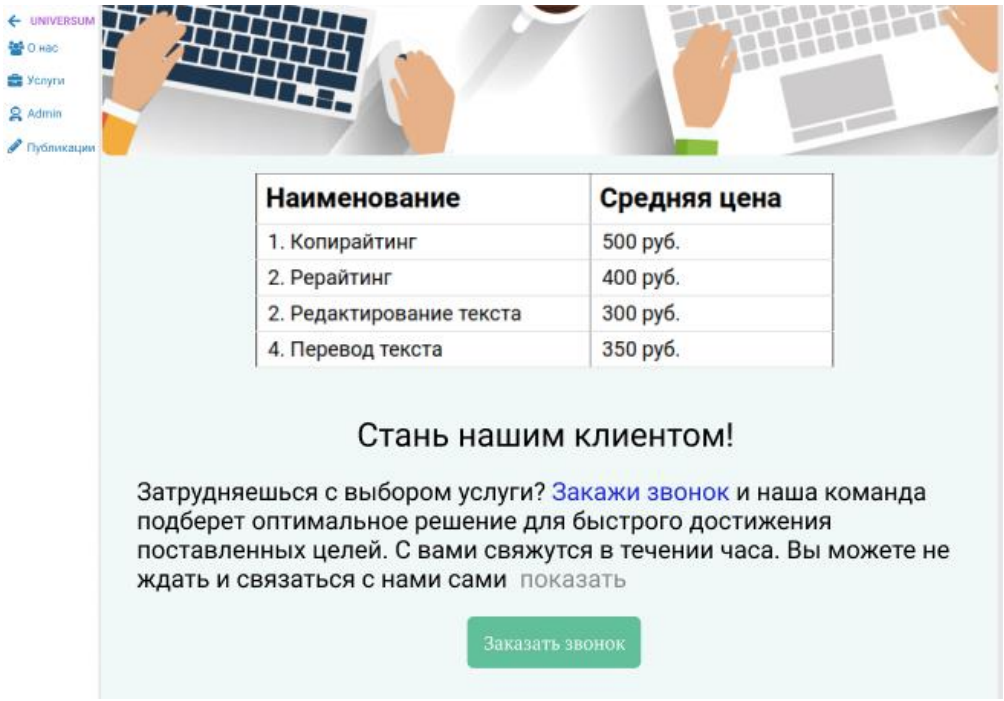

Рисунок 3.22 – Дизайн раздела услуг

8) Дизайн раздела всех публикаций (рисунок 3.23).

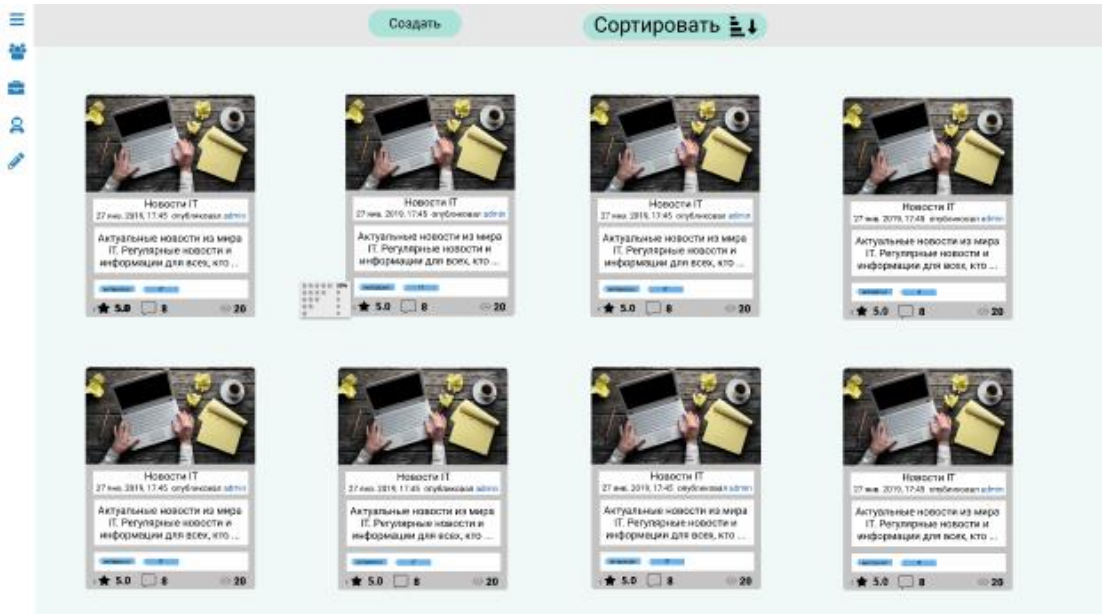

Рисунок 3.23 – Дизайн раздела всех публикаций

9) Дизайн раздела просмотра конкретной публикации (рисунок 3.24).

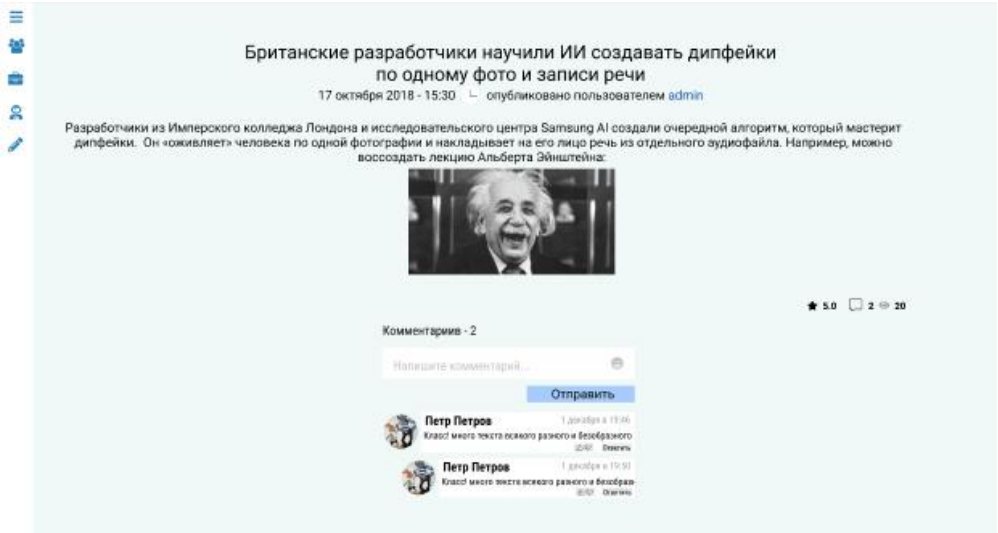

Рисунок 3.24 – Дизайн раздела просмотра конкретной публикации

10) Дизайн раздела создания публикации (рисунок 3.25).

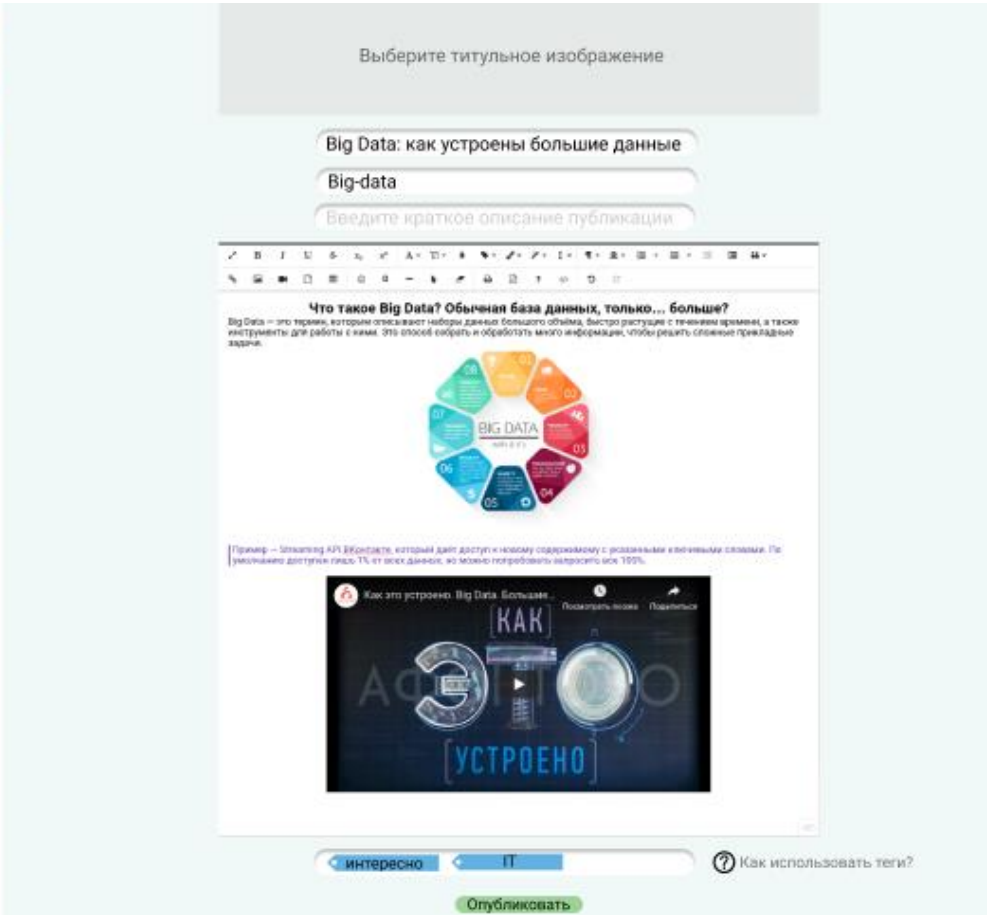

Рисунок 3.25 – Дизайн раздела создания публикации

11) Дизайн личного кабинета пользователя (рисунок 3.26).

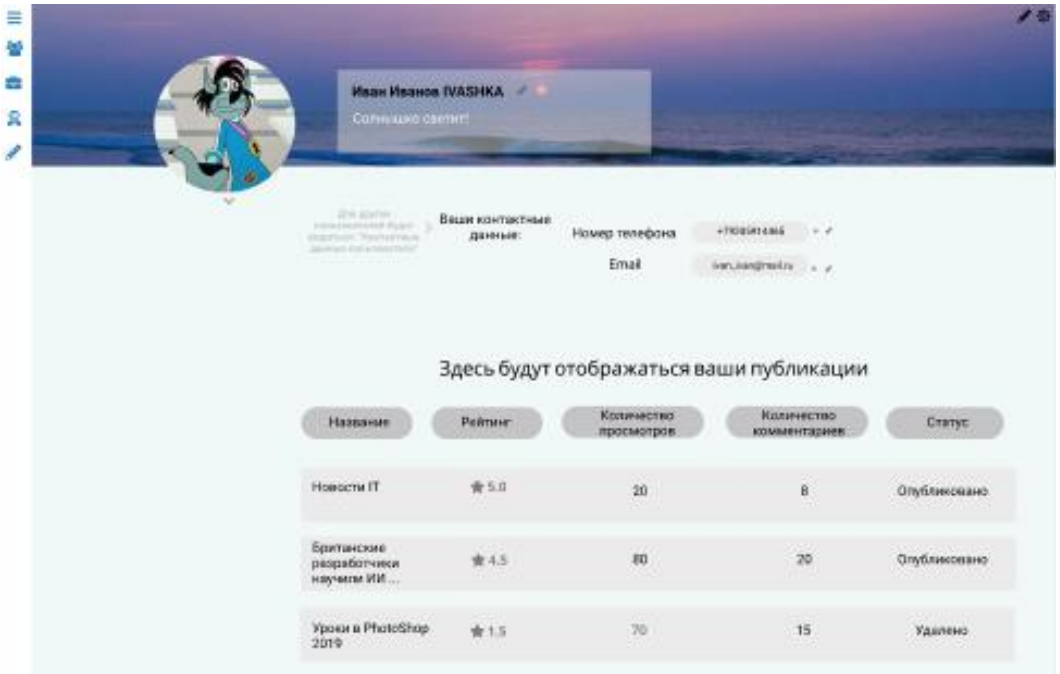

Рисунок 3.26 – Дизайн личного кабинета пользователя

12) Дизайн кабинета управления заказами для администраторов (рисунок 3.27).

Рисунок 3.27 – Дизайн личного кабинета пользователя

Далее идет этап процесса технической реализации – разработка базы данных. Разработку базы данных следует начать с выбора системы управления базами данных.

Проведем сравнительный анализ популярных СУБД (таблица 3.2).

Таблица 3.2 – Сравнительный анализ СУБД

Критерий	**Oracle DB**	**MySQL**	**PostgreSQL**
Тип	Мульти-модельная	Реляционная	Объектно-реляционная
Лицензия	Коммерческая	Свободная и коммерческая (для проектов, не желающих открывать свой исходный код)	Свободная
Исходный код	Закрытый	Открытый	Открытый
Стабильность	Высокая	Средняя	Высокая
Поддержка	Платная	Платная	Платная

В результате проведения сравнительного анализа, для разработки сайта была выбрана система управления базами данных PostgreSQL. Она распространяется по свободной лицензии, без каких-либо ограничений, имеет открытый исходный код и высокую стабильность.

На данном этапе была создана структура данных (рисунок 3.28) - выделены сущности, определены их свойства и связи между сущностями.

Структура данных – принцип или порядок организации записей в базе данных и связей между ними [25].

Ниже рассмотрим созданную структуру данных.

Таблица «posts» хранит в себе информацию о всех публикациях, представленных на сайте во вкладке «публикации». Таблица состоит из 11 полей: «id», «title», «title_image», «preview_content», «content», «created_at», «created_by», «updated_at», «updated_by», «publish_at», «views», «rating».

Таблица «users» хранит в себе информацию о всех зарегистрированных пользователях сайта. Таблица состоит из 9 полей: «id», «username», «first_name», «last_name», «image», «phone», «email», «password», «active».

Таблица «views» предназначена для регистрации всех просмотров публикаций. Состоит из 4 полей: «id», «post_id», «ip», «agent», «created_at».

Таблица «comments» предназначена для хранения всех комментариев к публикациям. Состоит из 6 полей: «id», «content», «created_at», «created_by», «post_id», «comment_parent».

Таблица «rating» предназначена для хранения оценок всех публикаций. Состоит из 6 полей: «id», «post_id», «rating», «created_at», «created_by», «comment».

Таблица «roles» предназначена для хранения всех ролей сайта. Состоит из полей: «id», «title», «description».

Таблица «user_roles» предназначена для осуществления связи «многие ко многим» таблиц «user» и «roles». Состоит из 2 полей: «user_id», «role_id».

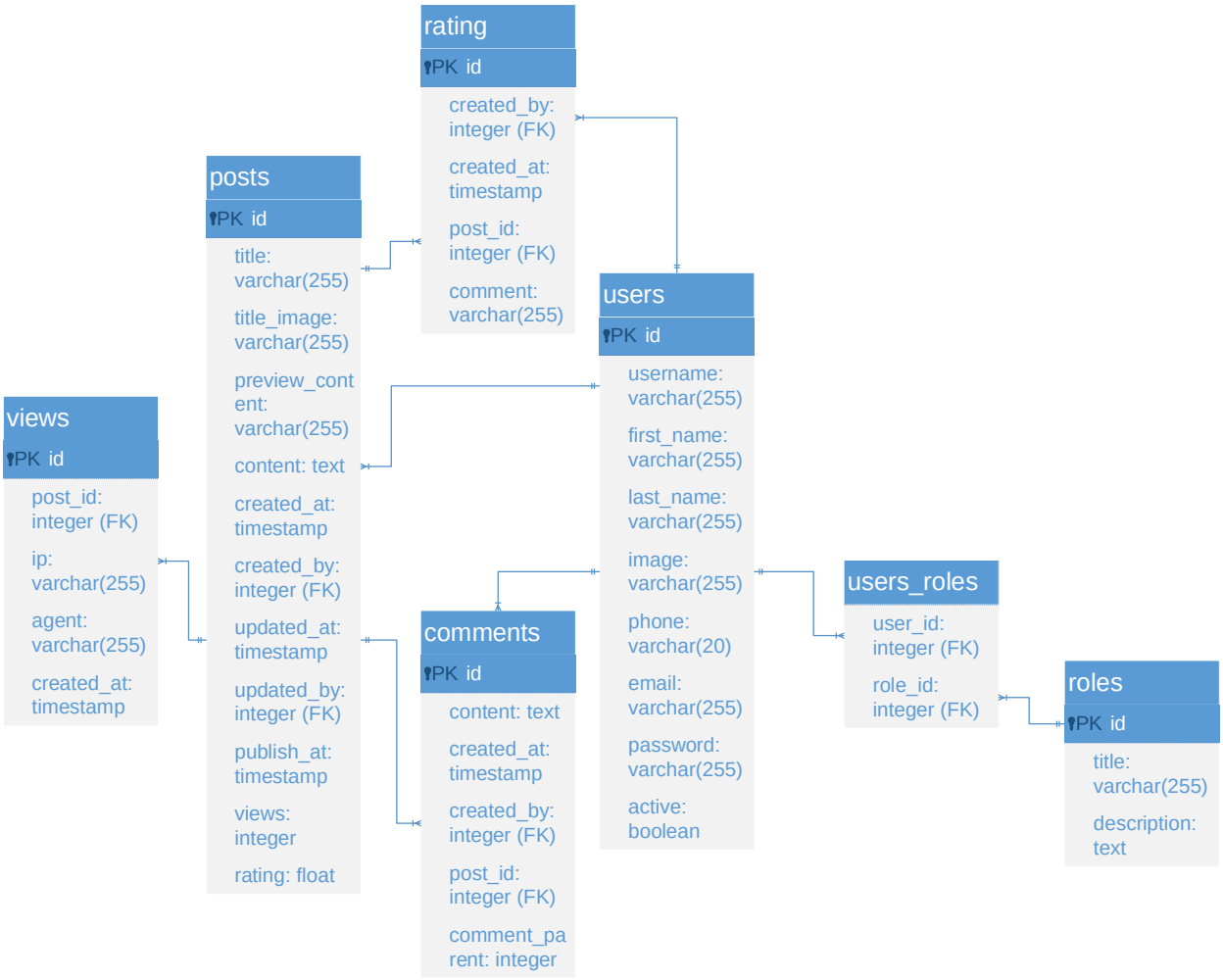

Рисунок 3.28 – Структура данных

Созданная структура данных соответствует третей нормальной форме.

Нормальная форма – это свойство отношения в реляционной модели данных, характеризующее его с точки зрения избыточности, потенциально приводящей к логически ошибочным результатам выборки или изменения данных. Нормальная форма определяется как совокупность требований, которым должно удовлетворять отношение. [26].

Представленная архитектура данных была реализована в выбранной ранее СУБД PostgreSQL.

Клиентская часть web-сайтов, в основном разрабатывается с помощью HTML, CSS и JavaScript. Причем у JavaScript существуют «фреймворки», структурирующие код и облегчающие процесс его написания.

Фреймворк (Framework) — это каркас приложения. Он обязывает разработчика выстраивать архитектуру приложения в соответствии с некоторой логикой [27]. Любой фреймворк обычно предоставляет функционал вроде событий, хранилищ и связывания данных.

Преимущества использования JavaScript фреймворков:

– эффективность – проекты, которые раньше заняли бы месяцы и сотни строчек кода, сейчас могут быть реализованы намного быстрее с хорошо структурированными готовыми шаблонами и функциями.

– безопасность – лучшие JavaScript фреймворки имеют фирменную систему безопасности и поддерживаются крупным сообществом.

– расходы – большинство фреймворков с открытым кодом и бесплатны. Поскольку они помогают программистам быстрее разрабатывать пользовательские решения, итоговая цена web приложения будет ниже.

В рейтинге «GitHub stars», который показывает общее количество «звезд», поставленных пользователями GitHub конкретному фреймворку, лидирующие позиции среди множества JavaScript инструментов для разработки клиентской части web-сайта занимают React и Vue.

На сегодняшний день GitHub – это один из самых популярных сервисов для совместной разработки программного обеспечения и его хостинга в «облаке» [28].

Рассмотрим подробнее популярные JavaScript фреймворки – React и Vue.

React и Vue во многом похожи. Они оба:

– используют концепцию Virtual DOM

– предоставляют реактивность и компонентную структуру

– фокусируются на корневой библиотеке, вынося прочие вопросы, такие как роутинг или управление глобальным состоянием приложения, в дополнительные библиотеки.

DOM (Document Object Model) – объектная модель документа, позволяющая считывать и менять содержимое, оформление и даже структуру html-документов [29].

Virtual DOM – это концепция программирования, в которой идеальное или «виртуальное» представление пользовательского интерфейса хранится в памяти и синхронизируется с «реальным» DOM [30].

Как React, так и Vue — исключительно быстрые, поэтому скорость вряд ли будет решающим фактором при выборе между ними.

В React когда состояние компонента изменяется, оно запускает повторную отрисовку всего поддерева компонента, начиная с себя.

Во Vue зависимости компонента автоматически отслеживаются во время отрисовки, поэтому система точно знает, какие компоненты действительно необходимо повторно отрисовывать при изменении состояния.

В React абсолютно всё — это JavaScript. Не только структуры HTML, выраженные через JSX, последние тенденции также включают управление CSS внутри JavaScript. Этот подход имеет свои преимущества, но также заставляет идти на компромиссы, которые могут показаться неэффективными.

JSX – это препроцессор, который добавляет синтаксис XML к JavaScript [31].

Vue охватывает классические веб-технологии и основывается на них, генерируя чистый HTML и CSS код.

Для организации модульности CSS в React используется подход «CSS-in-JS». Это представляет собой новый компонентно-ориентированный подход к стилизации, который отличается от обычного процесса разработки CSS.

Главным отличием между React и Vue здесь будет то, что по умолчанию стилизация Vue выполняется через CSS теги «style» в однофайловых компонентах.

В результате анализа приведенных выше JavaScript фреймворков, был выбран Vue, так как он оказался наиболее простым в освоении.

Далее, была реализована клиентская часть web-сайта, с использованием выбранного инструмента, а именно были созданы следующие компоненты:

– «Start» – компонент стартового раздела;

– «Sidebar» – компонент бокового навигационного меню;

– «AuthForm» – компонент модального окна регистрации и авторизации;

– «Main» – компонент основной части всех разделов сайта, в который помещается основная информация любого раздела;

– «Layout» – компонент, являющийся родительской «оберткой» для последующих компонентов, содержащий в себе компонент «Sidebar» и «Main»;

– «Index» – компонент главного раздела, обернут в компонент «Main»;

– «About» – компонент раздела «О нас», обернут в компонент «Main»;

– «Services» – компонент раздела «Услуги», обернут в компонент «Main»;

– «Comment» – компонент комментариев;

– «Post» – компонент публикации, содержит компонент «Comment» и обернут в компонент «Main»;

– «PostList» – компонент, содержащий список публикаций (компонентов Post), обернут в компонент «Main»;

– «PostEditor» – компонент создания и редактирования публикации, обернут в компонент «Layout»;

– «LK» – компонент личного кабинета пользователя, обернут в компонент «Main».

На рисунке 3.29 представлена диаграмма взаимосвязи созданных компонентов.

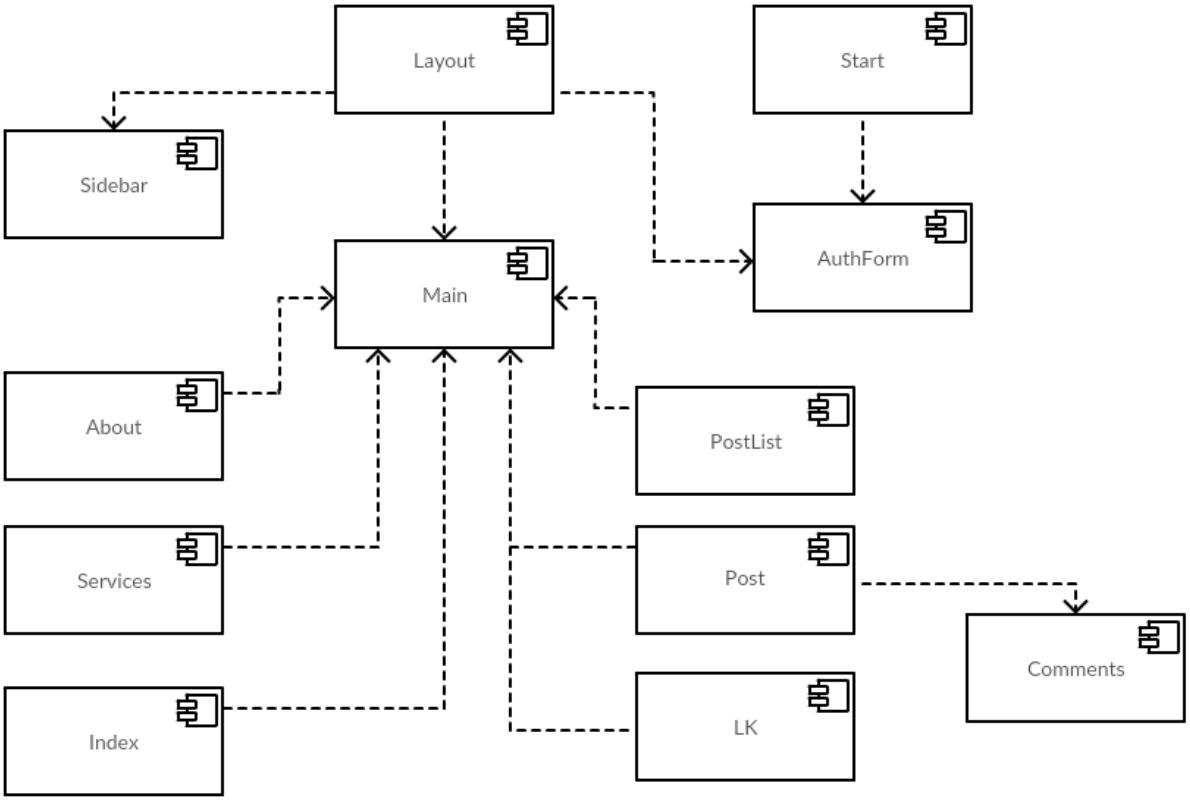

Рисунок 3.29 - Диаграмма взаимосвязи компонентов

В приложении А представлен листинг программного кода компонента «PostList».

Приступим к рассмотрению языков программирования серверной части web-сайтов. На сегодняшний день языков программирования, используемых для серверной web-разработки, существует большое количество: PHP, Ruby, Java, JavaScript, Python, Go, C# и многие другие.

В техническом плане для большинства проектов нет каких-либо ограничений при выборе языка, то есть почти любой функционал сайта может быть успешно реализован на любом из них, поэтому выбор языка не накладывает никаких лимитаций на проект.

Но с точки зрения производительности и экономической точки зрения разница есть. Рассмотрим языки программирования для серверной разработки web-приложений: PHP, JavaScript, Python, Go.

1) PHP – интерпретируемый язык программирования, исполняемый на стороне web-сервера, спроектированный Расмусом Лердорфом в 1995 году

в качестве инструмента создания динамических и интерактивных web-сайтов [32].

Этот язык оказался достаточно гибким и мощным, поэтому приобрёл большую популярность и используется в проектах любого масштаба: от простого блога до крупнейших web-приложений в Интернете.

К преимуществам PHP можно отнести:

– легкость в освоении на всех этапах;

– большое сообщество пользователей и разработчиков;

– имеет развитую поддержку баз данных;

– может использоваться в изолированной среде;

– предлагает встроенные средства организации web-сессий, программный интерфейс расширений;

– может быть развёрнут почти на любом сервере;

– портирован под большое количество аппаратных платформ и операционных систем.

К недостаткам PHP можно отнести:

– слабые средства для работы с исключениями;

– отсутствие асинхронности;

– глобальные параметры конфигурации влияют на базовый синтаксис языка, что затрудняет настройку сервера и разворачивание приложений;

В мировом рейтинге популярности языков программирования PYPL язык PHP занимает 5 место. Однако с 2008 года наблюдается непрерывный спад популярности языка.

Скрипты, написанные на языке PHP, обычно хранятся в файлах с расширением «.php», которые содержат в себе смесь обычных HTML-тэгов со специальной php разметкой. Исполнение таких файлов требуют наличия установленного на сервер интерпретатора языка PHP.

2) JavaScript – мультипарадигменный интерпретируемый язык программирования. Поддерживает объектно-ориентированный,

императивный и функциональный стили. Является реализацией языка ECMAScript (стандарт ECMA-262). Разработан в 1995 году Бренданом Эйхом [33].

К преимуществам JavaScript можно отнести:

– Популярность языка;

– Поддержка JavaScript всеми современными браузерами является стандартом де-факто и фактически не имеет альтернатив в разработке клиентской части web-сайтов;

– Асинхронность на уровне самого языка;

– Язык постоянно совершенствуется и развивается;

– Приведение типов данных проводится автоматически;

– Автоматическая очистка памяти или так называемая сборка мусора.

К недостаткам JavaScript можно отнести:

– Однопоточность языка;

– Сложность синтаксиса;

– Многословность.

В мировом рейтинге популярности языков программирования PYPL язык JavaScript занимает 3 место. Тенденция популярности на протяжении многих лет остается примерно на одном уровне.

3) Python широко применяется как интерпретируемый язык для скриптов различного назначения. Язык разработал Гвидо ван Россум в 1991 году.

Python имеет целью приблизить синтаксис реальной программы, написанной на нём, к описывающему задачу псевдокоду, что позволяет программисту уменьшить объём программы [34].

Элегантный дизайн и эффективный, дисциплинирующий синтаксис этого языка облегчают программистам совместную работу над кодом. Python – мультипарадигмальный язык программирования: он позволяет совмещать

процедурный подход к написанию кода с объектно-ориентированным и функциональным [35].

К преимуществам Python можно отнести:

– простота в изучении (особенно на начальном этапе);

– особенности синтаксиса стимулируют программиста писать хорошо читаемый код;

– предоставляет средства быстрого прототипирования и динамической семантики;

– имеет большое сообщество, позитивно настроенное по отношению к новичкам;

– имеет множество полезных библиотек и расширений языка, которые можно легко использовать в своих проектах благодаря предельно унифицированному механизму импорта и программным интерфейсам;

– механизмы модульности хорошо продуманы и могут быть легко использованы;

– абсолютно всё в Python является объектами в смысле ООП, но при этом объектный подход не навязывается программисту.

К недостаткам Python можно отнести:

– не слишком удачная поддержка многопоточности;

– изначальная ограниченность средств для работы с базами данных;

Python занимает 1 место среди популярных языков программирования по версии PYPL. Язык постоянно растет в своей популярности.

Для запуска программ, написанных на Python, нужен интерпретатор языка.

4) Go — строго типизированный, компилируемый, многопоточный язык программирования, разработанный внутри компании Google. Был выпущен в 2009 году, его непосредственным проектированием занимались Роберт Гризмер, Роб Пайк и Кен Томпсон [36].

К преимуществам Go можно отнести:

– Быстрая компиляция в машинный код;

– Скорость исполнения, по сравнению с интерпретируемыми языками программирования;

– Компилируется под все популярные ОС;

– Производительность и элегантный синтаксис;

– Многопоточность и ассинхронность «из коробки»;

– Предусмотрен автоматический сборщик мусора;

– Имеется большое количество библиотек.

К недостаткам Go можно отнести:

– Малая популярность

– Многословность

Golang занимает 14 место среди самых популярных языков программирования по версии PYPL. Язык непрерывно набирает популярность.

Развертка Go на web-сервере очень проста – необходимо лишь поместить скомпилированный файл на сервер и запустить его.

В ходе рассмотрения языков программирования для реализации серверной части web-сайта, был выбран язык Go. Программу, написанную на этом языке, очень легко запустить на сервере. А так как язык является компилируемым в машинный код, он гораздо быстрее любых интерпретируемых языков. Язык Go разрабатывался в качестве замены C: его высокая производительность почти сопоставима с языком C, но более простой синтаксис дает возможность разрабатывать приложения гораздо быстрее. Упрощенный синтаксис облегчает не только написание своего собственного кода, но и чтение кода, написанного другими программистами, что особенно важно в дальнейшем, когда к проекту присоединятся другие разработчики.

Далее на выбранном языке программирования Go был запрограммирован web-сервер, который выполняет функцию REST API-сервера.

REST (Representational state transfer) – это стиль архитектуры программного обеспечения для распределенных систем, который, как правило, используется для построения web-служб [37].

API – это программный интерфейс приложения, благодаря которому одна компьютерная программа может взаимодействовать с другой программой [38].

На языке Go было запрограммировано:

– API маршрутизация сайта;
– модели;
– контроллеры.

API маршрутизация предназначена для направления HTTP запроса к соответствующему контроллеру, с целью его обработки.

Модель (Model) – предоставляет собой объектную модель некой предметной области, включает в себя данные и методы работы с этими данными, реагирует на запросы из контроллера, возвращая данные и (или) изменяя своё состояние, при этом модель не содержит в себе информации, как данные можно визуализировать, а также не «общается» с пользователем напрямую [39].

Контроллер (Controller) обеспечивает связь между пользователем и системой, использует модель и представление для реализации необходимой реакции на действия пользователя, как правило, на уровне контроллера осуществляется фильтрация полученных данных и авторизация (проверяются права пользователя на выполнение действий или получение информации) [40].

В ходе программирования серверной части сайта были созданы следующие модели:

– «User» - представляет собой модель пользователя;
– «Group» - представляет собой модель группы пользователя;
– «Post» - представляет собой модель публикации;
– «Tag» - представляет собой модель ключевого слова публикации;

– «Comment» - представляет собой модель комментария.

На рисунке 3.30 представлена диаграмма классов разработанных моделей сайта.

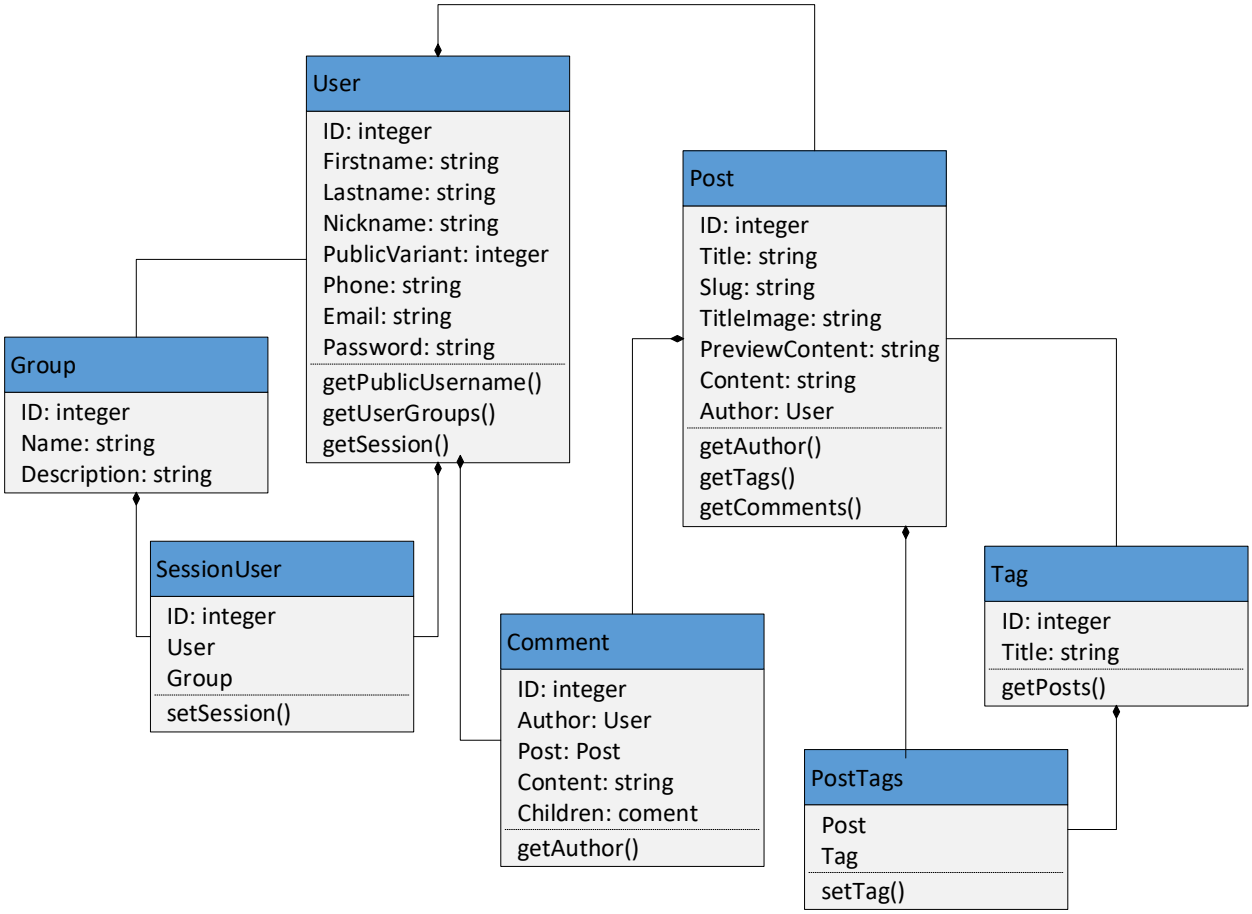

Рисунок 3.30 – Диаграмма классов моделей сайта

Также были созданы следующие контроллеры:

– «Service» - представляет собой контроллер услуг;

– «User» - представляет собой контроллер пользователей;

– «Post» - представляет собой контроллер публикаций;

– «Tag» - представляет собой контроллер ключевых слов публикаций;

– «Comment» - представляет собой контроллер комментариев.

Блок-схема алгоритма работы функции контроллера User, отвечающей за авторизацию пользователя на сайте, представлен на рисунке 3.31.

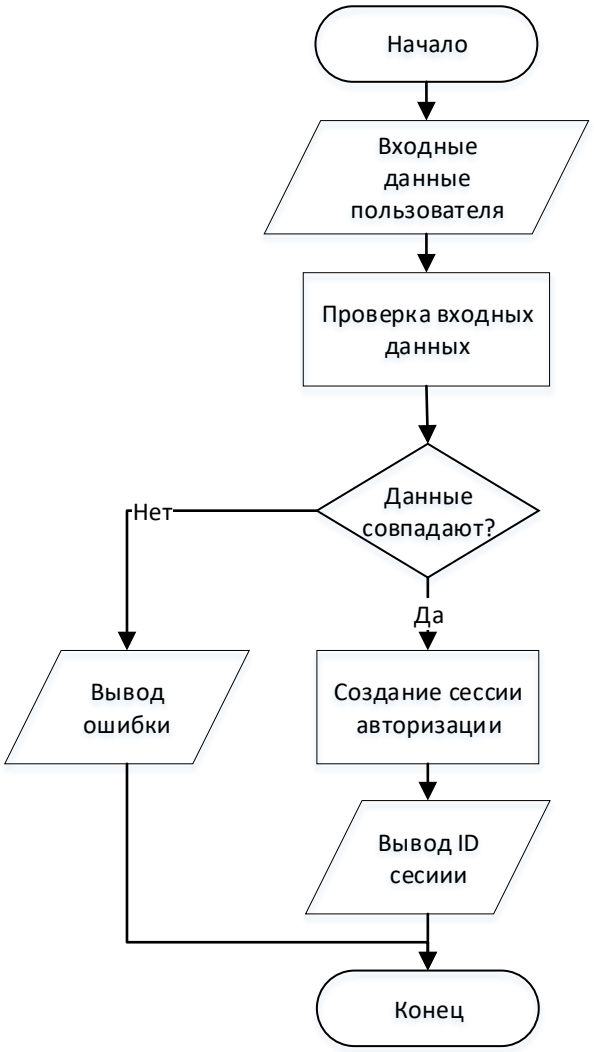

Рисунок 3.31 – Блок-схема алгоритма авторизации пользователя

В приложении Б представлен листинг программного кода функции контроллера «User», отвечающей за авторизацию пользователей на сайте.

3.3 Оценка экономической эффективности проекта

Процесс разработки web-сайта требует капиталовложений, поэтому при его создании возникает проблема оценки эффективности вложенного капитала. Целесообразность создания и функционирования web-сайта должна подтверждаться расчетами экономической эффективности.

Экономическая эффективность – это результат, который можно получить, соизмерив показатели доходности производства по отношению к общим затратам и использованным ресурсам [41].

Экономическая эффективность характеризуется показателями целесообразности всех затрат на его разработку и функционирование. При расчетах экономической эффективности сопоставляют затраты и результаты, а именно: затраты на разработку, создание и внедрение информационной системы, а также текущие затраты на ее эксплуатацию, с одной стороны, и, с другой стороны, результат – доход и сокращение прямых потерь ресурсов, получаемую в результате использования системы.

Рассмотрим затраты, связанные с созданием web-сайта.

1) Рассчитаем затраты на оплату труда разработчика web-сайта.

Расчет выполняется на основе трудоемкости выполнения каждого этапа в человеко-днях и величины среднемесячного оклада разработчика по городу Белгороду, вычисляется по формуле (3.1):

$$ЗП_м = О * Д, \quad (3.1)$$

где $ЗП_м$– оклад работника в месяц, руб.;

О – оклад работника в день, руб.;

Д – количество дней, затраченных разработчиком на создание web-сайта. Среднее количество рабочих дней в месяце равно 21. Следовательно, дневная заработная плата определяется путем разделения размера месячного оклада на среднее количество рабочих дней в месяце и вычисляется по формуле (3.2):

$$ЗП_д = \frac{М}{21}, \quad (3.2)$$

где $ЗП_д$– дневной оклад работника, руб.;

М – оклад работника в месяц, руб.;

Исходя из расчетов, оклад разработчика web-сайта за один рабочий день приведен в таблице 3.3.

Таблица 3.3 – Расчет оклада разработчика за один рабочий день

Должность	Среднемесячный оклад по городу Белгороду, руб	Стоимость одного рабочего дня, руб.
Разработчик	45 000	2 142

Исходя из результатов расчетов дневного оклада разработчика, определим расходы на выполнение всей работы. На выполнение своей работы разработчику понадобится 36 полных рабочих дней. Затраты на его заработную плату составляют 2142*36=77112 (руб.).

2) Рассмотрим затраты на рекламную компанию. За основу будет взята контекстная реклама от Yandex Директ, ее достоинством является возможность выставить собственную цену. Оптимальным решением будет выставить цену клика в 10 рублей, а дневным лимитом назначить сумму в 750 рублей. Таким образом, можно Yandex прогнозирует, что в день будет примерно 8000 – 12000 показов и 50 - 75 переходов на web-сайт. Выходит, что за месяц, организации необходимо вложить около 22500 рублей в рекламную компанию.

3) Рассмотрим затраты на аренду сервера. Для публикации сайта в сети Интернет подойдет самый простой VPS-сервер, арендованный у сервиса «reg.ru», стоимостью 2148 рублей в год.

4) Регистрация доменного имени в доменной зоне «ru» с помощью сервиса «reg.ru» обойдется в 199 рублей.

На аппаратное и программное обеспечение расходы не учитываются, так как при разработке web-сайта использовался собственный персональный компьютер с выходом в Интернет, а также ПО, распространяемое бесплатно.

Ниже приведена общая сумма необходимых затрат на создание и внедрение сайта для ООО «Раст». Таблица 3.4 отражает годовые затраты на поддержание web-сайта (включая рекламную компанию).

Таблица 3.4 – Затраты на годовое содержание web-сайта для ООО «Раст»

Наименование статьи затрат	Сумма, руб.
Покупка и регистрация доменного	199
Аренда сервера	2 148
Затраты на рекламную компанию	270 000
Итого:	272 347

Ежегодное содержание web-сайта для ООО «Раст» определяется путем суммирования всех затрат и составляет 272347 рублей.

Общие затраты составят 272347+77112=349459 рублей.

Рассмотрим целесообразность создания web-сайта для ООО «Раст». Для того чтобы узнать целесообразность, рассмотрим сколько времени тратит работник компании на поиск и консультацию клиентов. В среднем на поиск и информирование одного клиента работник ООО «Раст» тратит 25 минут рабочего времени. Сюда входит время на поиск клиента (15 минут) и консультацию (10 минут). Благодаря разработанному web-сайту и проведению планируемой рекламной компании оптимизируется процесс поиска и консультации клиентов, освобождая рабочее время сотрудников. Заинтересованный клиент, переходя на web-сайт самостоятельно ознакомится с услугами организации и их стоимостью. Далее клиент оставляет заявку на звонок или самостоятельно связывается с одним из сотрудников по номеру телефона. Консультация такого клиента, первично ознакомленного с необходимой информацией с помощью web-сайта, сократится и потребует у работника в среднем 5 минут рабочего времени.

Сравнивая время работы с одним клиентом, можно увидеть, что время, затрачиваемое на поиск и информирование одного клиента, сократится в 5 раз.

Рассмотрим, за какое время окупится разработка web-сайта для ООО «Раст».

На сегодняшний день у ООО «Раст» в среднем 50 клиентов в месяц. С помощью планируемого проведения рекламной компании, в среднем на сайт будет заходить в месяц около 400 посетителей, из которых, по статистике Yandex, реальных заказчиков около 30% от общего числа посетителей (120 человек), тем самым расширится клиентская база организации. По статистике, частные лица заказывают в ООО «Раст» услуги по написанию, переводу или редактированию текста, средняя стоимость которых 1000 рублей. Таким образом, ежемесячный доход составит 120*1000=120000 рублей (1440000 рублей в год). Также стоит учесть, что на сайте будут представлены собственные публикации, и материал, публикуемый зарегистрированными пользователями. В разделе публикаций будет подключена реклама от партнерской программы Google AdSense, так как подключить сайт к этой партнерской программе легче, чем к другим. При условии количества просмотров публикаций около 1000 в день, учитывая цену одноразового показа рекламы в 3 рубля и нажатия на рекламу – 6 рублей, получим примерный месячный доход равный 3120 рублей (3120*12=37440 рублей в год), при условии, что нажимать на рекламу будут около 2% пользователей. Совокупный годовой доход ООО «Раст» составит 1440000+37440=1477440 рублей. Итого, сайт окупится за 3 месяца (349459/1477440).

Таким образом, в данном разделе был проведен сравнительный анализ и выбор средств проектирования web-сайтов, спроектирован web-сайт для ООО «Раст», с помощью выбранных средств, а также проведена оценка экономической эффективности. После расчетов экономической эффективности было выявлено, что сайт окупится за 3 месяца. Учитывая время, которое работник тратит на поиск и консультацию одного клиента, а также короткий срок окупаемости web-сайта, можно утверждать о целесообразности создания web-сайта для ООО «Раст».

ЗАКЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была достигнута ее цель, а именно расширение клиентской базы организации.

Для достижения поставленной цели выпускной квалификационной работы были исследованы способы хранения и рейтингования публикаций, рассмотрены теоретические аспекты проектирования и разработки web-сайта. Исследована деятельность и проанализированы бизнес-процессы ООО «Раст».

Для реализации проекта были проанализированы и выбраны современные средства проектирования и разработки web-сайта, а именно:

– Figma – сервис, для создания прототипов и дизайн-макетов пользовательского интерфейса;

– PostgreSQL – система управления базами данных;

– Vue – фреймворк, для программирования клиентской части web-сайта;

– Go – язык программирования серверной части web-сайта.

Приведенные расчеты эффективности показывают, что последующее внедрение web-сайта обосновано. Разработанный проект имеет относительно короткий срок окупаемости 3 месяца и будет способствовать увеличению доходов ООО «Раст».

Результатом данного проекта является удобный web-сайт для ООО «Раст», удовлетворяющий требованиям заказчика и полностью готовый к применению. С его помощью пользователи могут получить необходимую информацию об организации, ознакомиться с предоставляемыми услугами, а также заказывать услуги. Также web-сайт позволяет пользователям просматривать публикации, зарегистрированным пользователям оценивать, комментировать и публиковать собственный материал. В свою очередь ООО «Раст» может монетизировать содержимое публикаций, путем подключения рекламы от сторонних партнерских программ.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Международное исследовательское агентство We Are Social [Электронный ресурс] - URL: https://wearesocial.com/blog/2018/10/the-state-of-the-internet-in-q4-2018 (дата обращения 25.03.2019)

2 Электронная база словарей, энциклопедий и книжных изданий Academic. Печатное издание [Электронный ресурс] - URL: https://dic.academic.ru/dic.nsf/ruwiki/300272 (дата обращения 25.04.2019)

3 Сетевое издание Adindex. Что происходит с печатной прессой в России и мире [Электронный ресурс] - URL: https://adindex.ru/publication/analitics/search/2018/07/26/172978.phtml (дата обращения 25.03.2019)

4 Свободная энциклопедия Википедия. Коллективный блог [Электронный ресурс] - URL: https://ru.wikipedia.org/wiki/Коллективный_блог (дата обращения 25.03.2019)

5 Образовательный журнал Tilda. Социальные сети [Электронный ресурс] - URL: http://tilda.education/courses/marketing/social-networks/ (дата обращения 25.03.2019)

6 Венедюхин, А. Создание сайтов [Текст] / А. Вендюхин, А. Воробьев. - М.: Эксмо, 2016. - 528 с.

7 Лавдей, Л. Проектирование прибыльных веб-сайтов [Текст] / Лавдей, Л, Нихаус С. – М.: Эксмо, 2014. - 256 с.

8 Academic. Концепция [Электронный ресурс] - URL: https://dic.academic.ru/dic.nsf/enc_philosophy/2537/концепция/ (дата обращения 26.03.2019)

9 Фридман, В. А. Строительство Web-сайтов [Текст] / В. А. Фридман, А. В. Александров. - М.: Триумф, 2015. - 288 с.

10 Semantica. Прототип сайта [Электронный ресурс] - URL: https://semantica.in/blog/sajta-prototip-eto-chto-za-tip.html (дата обращения 28.03.2019)

11 Дакетт, Д. HTML и CSS. Разработка и дизайн веб-сайтов [Текст] / Д. Дакетт. – М.: Эксмо, 2013. - 480 с.

12 Дронов, В. HTML 5, CSS 3 и Web 2.0. Разработка современных Web-сайтов [Текст] / В. Дронов. – СПБ.: БХВ- Петербург, 2014. - 138 с.

13 Роббинс, Д. HTML5. Карманный справочник [Текст] / Д. Роббинс. - М.: Вильямс, 2019. - 192 с.

14 Голицына, О.Л. Базы данных [Текст] / О.Л. Голицына, Н.В. Максимов, И.И. Попов. - М.: Форум, 2012. - 400 с.

15 Кузин, А.В. Базы данных: Учебное пособие для студ. высш. учеб. заведений [Текст] / А.В. Кузин. - М.: ИЦ Академия, 2014. – 320 с.

16 Осипов, Д. Технологии проектирования баз данных [Текст] / Д. Осипов. – М.: ДМК Пресс, 2019 – 498 с.

17 Осипов, Д. Основы технологий баз данных [Текст] / Д. Осипов. – М.: ДМК Пресс, 2019 – 240 с.

18 Academic. Клиент-серверная СУБД [Электронный ресурс] - URL: https://dic.academic.ru/dic.nsf/ruwiki/144036/Клиент (дата обращения 01.04.2019)

19 Academic. Встраиваемая СУБД [Электронный ресурс] - URL: https://dic.academic.ru/dic.nsf/ruwiki/1771930 (дата обращения 01.05.2019)

20 Уильямс, Б. WordPress для профессионалов. Разработка и дизайн сайтов [Текст] / Б. Уильямс, Д. Дэмстра. - М.: Вильямс, 2014. – 464 с.

21 W3techs. Использование систем управления контентом [Электронный ресурс] - URL: https://w3techs.com/technologies/overview/content_management/all (дата обращения 01.04.2019)

22 Reghelp. ЕГРЮЛ [Электронный ресурс] - URL: http://www.reghelp.ru/egrul.shtml (дата обращения 02.05.2019)

23 Megaidei. Что такое ОКВЕД [Электронный ресурс] - URL: http://megaidei.ru/organizaciya-biznesa/chto-takoe-okved (дата обращения 02.04.2019)

24 Википедия. Модальное окно [Электронный ресурс] - URL: https://ru.wikipedia.org/wiki/Модальное_окно (дата обращения 05.04.2019)

25 Academic. Структура базы данных [Электронный ресурс] - URL: https://dic.academic.ru/dic.nsf/fin_enc/30016 (дата обращения 06.04.2019)

26 Шегин, Г. PostgreSQL 11. Мастерство разработки. Как специалисты создают масштабируемые, надежные и отказоустойчивые приложения базы данных [Текст] / Г. Шегин. - М.: ДМК Пресс, 2019 – 352 с.

27 Глоссарий интернет-маркетинга. Что такое Framework [Электронный ресурс] - URL: http://www.glossary-internet.ru/terms/F/3201/ (дата обращения 12.04.2019)

28 VScode. GitHub – что это [Электронный ресурс] – URL: https://vscode.ru/articles/github-chto-eto.html (дата обращения 13.04.2019)

29 Proglib. Объектная модель документа: что такое DOM [Электронный ресурс] - URL: https://proglib.io/p/what-is-dom/ (дата обращения 15.04.2019)

30 Чиннатамби, К. Изучаем React [Текст] / К. Чиннатамби. – М.: Эксмо, 2019 – 368 с.

31 Thewebland. Введение в JSX [Электронный ресурс] - URL: https://thewebland.net/development/javascript/reactjs/introducing-jsx/ (дата обращения 17.04.2019)

32 Симдянов, И. PHP 7. В подлиннике [Текст] / И. Симдянов, Д. Котеров. – М.: БХВ-Петербург, 2016 – 1073 с.

33 Флэнган, Д. JavaScript. Подробное руководство [Текст] / Д. Флэнган. – М.: Символ-Плюс, 2014 – 992 с.

34 Шоу З. Лёгкий способ выучить Python [Текст] / З. Шоу. – М.: Бомбора, 2017 г. – 352 с.

35 Жуков Р.А. Язык программирования Python [Текст] / Р. А. Жуков. – М.: НИЦ ИНФРА-М, 2019. – 216 с.

36 Джепикс, Ф. Язык программирования Golang [Текст] / Ф. Джепикс, Э. Троелсен. – М.: Вильямс, 2016 – 1440 с.

37 Harb. Архитектура REST [Электронный ресурс] - URL: https://habr.com/ru/post/38730/ (дата обращения 20.04.2019)

38 Techrocks. Что такое API [Электронный ресурс] - URL: https://techrocks.ru/2018/05/29/api-simple-explanation/ (дата обращения 25.04.2019)

39 Web-creator. MVC – модель-представление-контроллер [Электронный ресурс] - URL: https://web-creator.ru/articles/mvc (дата обращения 27.04.2019)

40 Tproger. MVC: что это такое и какое отношение имеет к пользовательскому интерфейсу [Электронный ресурс] - URL: https://tproger.ru/articles/mvc/ (дата обращения 27.04.2019)

41 Алексахин, А. Управление проектами и экономическая эффективность. Оценка экономической эффективности проектов [Текст] / А. Алексахин А. Ю. Костюхин, О. Скрябин. – М.: Вильямс, 2015. - 67 с.

ПРИЛОЖЕНИЕ А

```
<template>
  <layout>
    <div class="header-postlist">
      <router-link v-if="isAuth" :to="{name: 'postCreate'}" class="auth-
button">Создать</router-link>
      <div v-else @click="goLogin" class="auth-button">Авторизоваться</div>
      <base-select v-model="sortBy" :selectAttrs="{ items }"
                   @input="changeSort"/>
    </div>
    <div class="awrapper">
      <post v-for="post in parsedPosts" :key="post.id" :post="post"
@deletePost="deletePost"></post>
      <infinite-loading @infinite="infiniteLoadPosts" :distance="500"
ref="infiniteLoading">
        <div slot="no-more" style="color: grey; padding: 15px">Кажется, это
все (:</div>
        <div slot="no-results" style="color: grey; padding: 15px">Ничего не
найдено :(</div>
      </infinite-loading>
    </div>
  </layout>
</template>

<script>
  import axios from 'axios';
  import Layout from "../layout";
  import post from "./post";
  import InfiniteLoading from 'vue-infinite-loading';
  import moment from 'moment'
  import Post from "@/classes/post";
  import isMobileMixin from "../../mixins/isMobileMixin";
  import BaseSelect from "../mini/baseSelect";
  export default {
    name: "articles",
    mixins: [isMobileMixin],
    components: {
      BaseSelect,
      Layout,
      post,
      InfiniteLoading,
    },
    data() {
      return {
        posts: [],
        nextCount: 1,
        timestamp: "",
        loading: false,
        sortBy: "newer",
        items: [
          {value: "newer", text: "Новизне"},
          {value: "older", text: "Старине"},
```

```
      {value: "views", text: "Просмотры"},
      {value: "best", text: "Лучшие"}
    ]
  }
},
computed: {
  isAuth() {
    return this.$store.getters.isAuth
  },
  parsedPosts() {
    return Post.parsePostList(this.posts);
  },
},
methods: {
  deletePost(id) {
    let index = this.posts.map(item => item.id).indexOf(id);
    this.posts.splice(index, 1);
  },
  infiniteLoadPosts($state) {
    if (this.nextCount > 0) {
      axios({
        url: 'posts/',
        method: 'get',
        params: {
          date: this.timestamp,
          sort_by: this.sortBy,
          views: 9999,
        }
      }).then((response => {
        for (let p of response.data.data) {
          this.posts.push(p);
        }
        if (response.data.meta.next) {
          this.timestamp = response.data.meta.next;
        }
        if (response.data.meta.views) {
          this.views = response.data.meta.views
        }

        this.nextCount = response.data.meta.next_count;
        if (response.data.data.length > 1) {
          $state.loaded();
        } else {
          $state.complete();
        }
      })).catch((error) => {
        this.nextCount = 0;
      })
    } else {
      $state.complete();
    }
  },
```

```
      changeSort() {
        let now = moment().format("YYYY-MM-DTHH:mm:ss");
        if (this.sortBy === 'newer') {
          this.posts = [];
          this.timestamp = now
        } else if (this.sortBy === 'older') {
          this.posts = [];
          this.timestamp = "1970-01-01T00:00:01";
        }  else if (this.sortBy === 'views') {
          this.posts = [];
          this.timestamp = now;
        } else if (this.sortBy === 'best') {
          this.posts = [];
          this.timestamp = now;
        }
        this.nextCount = 1;
        this.$refs.infiniteLoading.stateChanger.reset()
      },
      goLogin() {
        this.$store.dispatch('changeShowLoginForm')
      },
    },
    created() {
      this.timestamp = moment().format("YYYY-MM-DTHH:mm:ss")
    }
  }
</script>
```

ПРИЛОЖЕНИЕ Б

```
func Login(w http.ResponseWriter, r *http.Request) {
    var message []string
    var valid bool
    var creds Credentials
    var user models.User
    code := api.StatusCode["ErrAuth"]
    valid = true
    err = json.NewDecoder(r.Body).Decode(&creds)
    if err != nil {
        fmt.Println(err)
        message = append(message, "Что-то пошло не так")
        valid = false
        w.WriteHeader(http.StatusInternalServerError)
    }
    if creds.Login == "" || creds.Password == "" {
        message = append(message, "логин или пароль пустой")
        valid = false
        w.WriteHeader(http.StatusBadRequest)
    }
    var username models.Username
    if valid {
        err = db.Db.QueryRow(`SELECT id, nickname, first_name, last_name,
phone, email, password, variant
                        FROM users WHERE email = $1 OR phone = $1`,
creds.Login).Scan(&user.ID, &username.Nickname,
            &username.FirstName, &username.LastName, &user.Phone,
&user.Email, &user.Password, &username.Variant)
        if err != nil {
            println(err)
        }
        equal := helpers.ComparePasswords(user.Password, creds.Password)
        if !equal {
            message = append(message, "Неверный логин/пароль")
            valid = false
            w.WriteHeader(http.StatusBadRequest)
        }
    }
    randomUuid, _ := uuid.NewV4()
    sessionToken := randomUuid.String()
    expires := 0
    if creds.Remember {
        expires = 3600
    }
    var groups []models.Group
    rows, _ := db.Db.Query(`SELECT groups.id, groups.name, groups.description
FROM groups INNER JOIN user_groups ON groups.id = user_groups.group_id WHERE
user_groups.user_id = $1`, user.ID)
    for rows.Next() {
        var g models.Group
        err = rows.Scan(&g.ID, &g.Name, &g.Description)
```

```
            if err != nil {
                println(err)
            }
            groups = append(groups, g)
        }
        groups2, _ := json.Marshal(groups)
        _, err = db.Cache.Do("HMSET", sessionToken, "user_id", user.ID, "groups",
groups2)
        _, err = db.Cache.Do("EXPIRE", sessionToken, expires)
        if err != nil {
            println(err)
            message = append(message, "Что-то пошло не так")
            valid = false
            w.WriteHeader(http.StatusInternalServerError)
            return
        }
        if valid {
            if creds.Remember {
                http.SetCookie(w, &http.Cookie{
                    Name:     "session_id",
                    Value:    sessionToken,
                    Path:     "/",
                    Expires:  time.Now().Add(time.Duration(expires)*time.Second),
                })
            } else {
                http.SetCookie(w, &http.Cookie{
                    Name:     "session_id",
                    Value:    sessionToken,
                    Path:     "/",
                })
            }
            message = append(message, "Вход выполнен успешно")
            var au models.AuthUser
            au.Id = user.ID
            au.Username = helpers.PublicUsername(username)
            if len(groups) > 0 {
                models.SetGroups(&au, groups)
            } else {
                au.Groups = []models.Group{}
            }
            json.NewEncoder(w).Encode(api.Response{
                true,
                &au,
                nil})
        } else {
            json.NewEncoder(w).Encode(api.Response{
                false,
                api.ErrMessage{&code, &message},
                nil})
        }
    }
```