

**Содержание выпускной квалификационной работы**

Введение

1 Нормативные ссылки

2 Термины, определения и сокращения

3 Анализ предметной области

4 Постановка задачи и техническое задание

5 Обзор используемых технологий и сред разработки

6 Проектирование базы данных

7 Реализация работы сервера с API

8 Тестирование программной платформы

9 Руководство пользователя

10 Безопасность и экологичность проекта

Заключение

Список использованных источников

Общее количество листов ПЗ 58

**Объем иллюстративной части**

1. Основная концепция проекта.

2. Серверная часть.

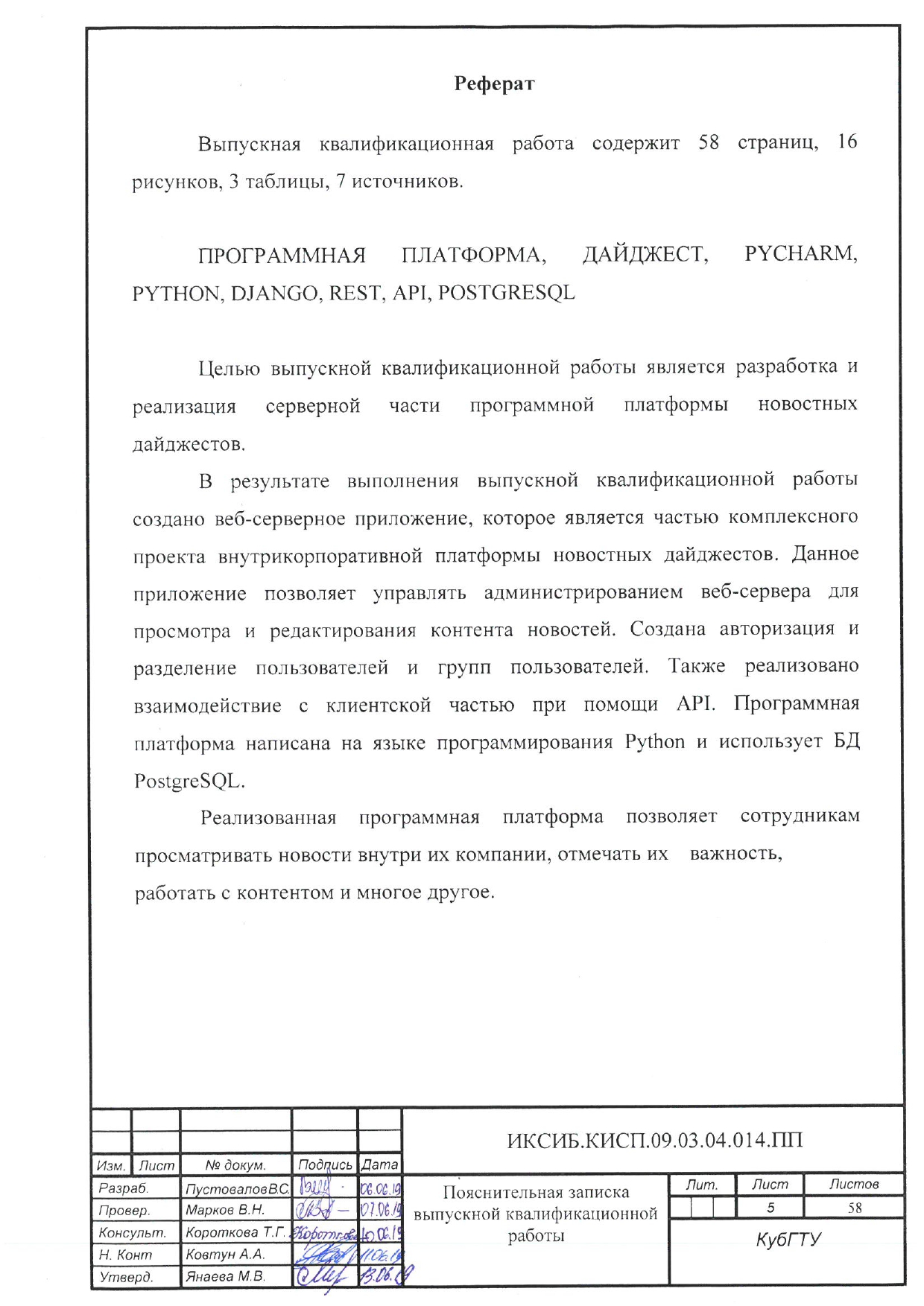
3. Клиентская часть.

4. Использование системы

Общее количество слайдов иллюстративной части 16

**Список основной и рекомендуемой литературы**

1. Уильям С. Винсент. Django для начинающих: создание сайтов на Python и Django, 1-е издание. – Пер. с англ. – СПб : Символ-Плюс, 2018. – 344 с



Содержание

[Введение 8](#_Toc11288474)

[1 Нормативные ссылки 9](#_Toc11288475)

[2 Термины, определения и сокращения 10](#_Toc11288476)

[3 Анализ предметной области 13](#_Toc11288477)

[4 Постановка задачи и техническое задание 15](#_Toc11288478)

[4.1 Общие требования 15](#_Toc11288479)

[4.2 Платформа 15](#_Toc11288480)

[4.3 Регистрация и авторизация 16](#_Toc11288481)

[4.4 Дайджесты 17](#_Toc11288482)

[4.5 Новости 18](#_Toc11288483)

[4.6 Новости с картинкой 18](#_Toc11288484)

[4.7 Текстовые новости 19](#_Toc11288485)

[4.8 Главные новости 19](#_Toc11288486)

[4.9 Новости сотрудников 19](#_Toc11288487)

[4.10 Профили 19](#_Toc11288488)

[4.11 Новости проектов 20](#_Toc11288489)

[5 Обзор используемых технологий и сред разработки 21](#_Toc11288490)

[5.1 Язык программирования Python 21](#_Toc11288491)

[5.2 JetBrains PyCharm Educational 2019 22](#_Toc11288492)

[5.3 Django Web Framework 23](#_Toc11288493)

[5.4 Django Rest Framework 24](#_Toc11288494)

[5.5 PostgreSQL 25](#_Toc11288495)

[6 Проектирование базы данных 27](#_Toc11288496)

[6.1 Проектирование концептуальной модели базы данных 27](#_Toc11288497)

[6.2 Проектирование логической модели базы данных 28](#_Toc11288498)

[6.3 Проектирование реляционной модели базы данных 29](#_Toc11288499)

[6.4 Физическое проектирование базы данных 32](#_Toc11288500)

[7 Реализация работы сервера с API 37](#_Toc11288501)

[7.1 Сериализация 37](#_Toc11288502)

[7.2 Представления 39](#_Toc11288503)

[7.3 Маршрутизация 40](#_Toc11288504)

[8 Тестирование программной платформы 42](#_Toc11288505)

[9 Руководство пользователя 45](#_Toc11288506)

[10 Безопасность и экологичность проекта 51](#_Toc11288507)

[10.1 Значение и задачи безопасности жизнедеятельности 51](#_Toc11288508)

[10.2 Анализ условий труда и мероприятия по защите от воздействия вредных производственных факторов 52](#_Toc11288509)

[10.3 Обеспечение электробезопасности 55](#_Toc11288510)

[10.4 Пожарная безопасность 55](#_Toc11288511)

[Заключение 57](#_Toc11288512)

[Список использованных источников 58](#_Toc11288513)

# Введение

В современном мире человечество всё более активно использует Интернет для работы и получения информационного контента. Благодаря этому в последнее время наблюдается быстрое развитие и совершенствование различных веб-сервисов, которые предлагают самые современные возможности, и сложные приложения, делающие повседневную жизнь значительно проще.

В связи с этим, возникла идея создания информационной системы, с помощью которой сотрудники смогут обмениваться информацией, узнавать самые последние новости внутри своего предприятия. Подобная технология была бы очень востребована в различных компаниях. Так появился проект DiDi – программная платформа новостных дайджестов.

DiDi является системой, позволяющей пользователям просматривать новостной контент, добавлять его в список важных, узнавать самые последние актуальные новости и быть в курсе всех событий, происходящих в своей компании. Также сотрудники могут просматривать различные договора, реквизиты и использовать их для удобной работы на предприятии. В рамках данной выпускной квалификационной работы требовалось спроектировать и реализовать серверную часть программной платформы новостных дайджестов DiDi.

DiDi проектировался, как веб-сервис. В настоящее время веб-приложения являются более доступными для пользователей в отличии от настольных. Разработанная программная платформа ориентирована на крупные компании и большое количество сотрудников.

Целью выпускной квалификационной работы являлась разработка информационного веб-сервиса для упрощения работы и удобного информирования сотрудников различных компаний.

# 1 Нормативные ссылки

В настоящей выпускной квалификационной работе использованы ссылки на следующие нормативные документы:

1. ГОСТ Р 1.12-2004. Стандартизация в Российской Федерации;

2. ГОСТ 20886-85 Организация данных в системах обработки данных. Термины и определения.

3. ГОСТ 34.321-96 Информационные технологии. Система стандартов по базам данных. Эталонная модель управления данными.

4. ГОСТ 15971-90 Системы обработки информации. Термины и определения.

5. СанПиН 2.2.2/2.4.1340 03 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы.

6. Трудовой кодекс Российской Федерации от 30.12.2001 № 197-ФЗ (ред. от 10.01.2016).

7. ТОИ Р 45 084 01 Типовая инструкция по охране труда при работе на персональном компьютере.

8. ГОСТ 12.0.003 2015 Система стандартов безопасности труда. Опасные и вредные производственные факторы. Классификация.

9. ГОСТ Р 2.2.2006 05 Руководство по гигиенической оценке факторов рабочей среды и трудового процесса. Критерии и классификация условий труда.

# 2 Термины, определения и сокращения

В настоящей выпускной квалификационной работе применяются термины с соответствующими определениями и сокращениями, установленные следующими нормативными документами:

– ГОСТ Р 1.12-2004. Стандартизация в Российской Федерации;

– ГОСТ 34.321-96 Информационные технологии. Система стандартов по базам данных. Эталонная модель управления данными;

– ГОСТ 15971-90 Системы обработки информации. Термины и определения;

– ГОСТ 20886-85 Организация данных в системах обработки данных.

Термины, определения и сокращения:

**1 Дайджест** – краткое содержание других публикаций.

**2 Язык программирования (ЯП)** – формальный язык, предназначенный для записи компьютерных программ.

**3 Электронно-вычислительная машина (ЭВМ**) – комплекс технических, аппаратных и программных средств, предназначенных для автоматической обработки информации, вычислений, автоматического управления. При этом основные функциональные элементы (логические, запоминающие, индикационные и др.) выполнены на электронных элементах.

**4 Персональная электронно-вычислительная машина (ПЭВМ)** – настольная микро-ЭВМ, имеющая эксплуатационные характеристики бытового прибора и универсальные функциональные возможности.

**5 База данных (БД)** – совокупность взаимосвязанных данных, организованных в соответствии со схемой базы данных таким образом, чтобы с ними мог работать пользователь.

**6 Система управления базами данных (СУБД)** – совокупность программных и языковых средств, обеспечивающих управление базами данных.

**7 Реляционная модель данных (РМД)** – логическая модель данных, прикладная теория построения баз данных, которая является приложением к задачам обработки данных таких разделов математики, как теория множеств и логика первого порядка.

**8 Реляционная база данных (РБД)** – база данных, реализованная в соответствии с реляционной моделью данных.

**9 Structured Query Language (SQL)** – декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

**10 Uniform Resource Locator (URL)** – система унифицированных адресов электронных ресурсов, или единообразный определитель местонахождения ресурса (файла).

**11 HyperText Transfer Protocol (HTTP)** – протокол прикладного уровня, используется передачи произвольных данных.

**12 HyperText Transfer Protocol Secure (HTTPS)** – расширение протокола HTTP для поддержки шифрования в целях безопасности.

**13 Integrated Development Environment (IDE)** – система программных средств, используемая программистами для разработки программного обеспечения.

**14 Application Programming Interface (API)** – описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой.

**15 Фреймворк** – заготовки, шаблоны для программной платформы, определяющие архитектуру программной системы; программное обеспечение, облегчающее разработку и объединение разных модулей программного проекта.

**16 Сериализация** – процесс перевода какой-либо структуры данных в последовательность битов.

# 3 Анализ предметной области

Дайджест представляет собой краткое изложение какой-либо информации. Также дайджестом может называться различная подборка выдержек, написанных по одной тематике и собранных из разных источников. Дайджест создан для того, чтобы можно было проще прочитать информацию, ведь новость может содержать достаточно большой объём текста, а времени на её изучение просто нет. Причём правильно составленная сжатая информация хорошо отображает структуру информации из источника, а также доходчиво излагает факты и человеку после прочтения, возможно, даже не придётся открывать оригинал.

На данный момент можно выделить четыре главные группы дайджестов:

– печатный дайджест;

– электронный дайджест, который может быть версией печатного дайджеста;

– Интернет-дайджест, который не имеет печатной версии;

– теледайджест, видеодайджест, а также радиодайджест.

Также дайджесты делятся на несколько типов:

1. Журналы. Они могут быть как в электронной, так и в печатной версии. Журналом называют сборник сжатых резонансных материалов, которые были опубликованы в других статьях или журналах.

2. Сборники аннотаций к книгам. Они пишутся для того, чтобы читатель не тратил своё личное время на полное чтение книги, а ознакомился с главной мыслью в концентрированном виде.

3. Новостные дайджесты. Как правило это могут быть различные новости в сокращённом виде, подборки свежих и актуальных новостей, события дня или недели. Также могут разделяться на новости в сфере экономики, образования, политики и так далее.

4. Обзоры периодических изданий. Обычно после выхода каких-либо газет, журналов или ежегодников, которые выходят с заявленной периодичностью, могут создаваться обзоры, в которых описывается главная суть.

5. Видео. Самое главное – это то, что видео посвящены каким-либо событиям, играм, кино и в них излагается основная мысль.

Для разработки программного обеспечения был выбран такой тип, как новостной дайджест. Для того, чтобы добиться поставленной задачи, необходимо разработать информационный портал новостных дайджестов. Информационные порталы делятся на несколько групп:

1. Горизонтальный портал. Это портал, который охватывает много различных тем.

2. Вертикальный портал. Как правило, это портал, который специализируется по определённой тематике.

Также порталы подразделяются на:

– региональные;

– интернациональные;

– публичные;

– корпоративные.

«В настоящее время в Интернете существует огромное количество дайджестов, предназначенных для самой широкой целевой аудитории. Прежде всего, это различные дайджесты прессы, ориентированные на определенную тематику: экономическую, политическую, культурную, спортивную, музыкальную и т. д.» [1, 50]

Программные платформы новостных дайджестов – это информационные порталы, в которых могут заинтересоваться различные предприятия, так как на портале могут быть размещены разные новости, баннеры, ссылки или реклама. Соответственно был сделан выбор в пользу корпоративной платформы новостных дайджестов.

# 4 Постановка задачи и техническое задание

Для понимания сути проектирования и разработки программного продукта необходимо правильно сформулировать постановку задачи. Постановка задачи содержит описание входной и выходной информации, а также формулировку технического задания.

Техническое задание содержит необходимые требования к программному продукту, выбор технологий и платформ для разработки веб-сервера, а также подробное описание объектов проектирования.

4.1 Общие требования

Серверная часть программной платформы новостных дайджестов предназначена для сотрудников предприятия и должна обеспечивать:

– постоянный доступ на внутрикорпоративный сайт новостей;

– возможность авторизации и аутентификации пользователя;

– возможность управления администрированием базы данных;

– добавление данных через администрирование;

– изменение данных через администрирование;

– удаление данных через администрирование;

– возможность индивидуальных пометок важных новостей для каждого пользователя;

– отправление GET-ответов с необходимыми данными в формате JSON при запросах с клиентской части.

4.2 Платформа

Сайт должен быть разработан на платформе Django Framework. Это свободно распространяющийся фреймворк, через который очень удобно разрабатывать подобные веб-сервисы. С помощью этого простого и современного фреймворка можно создавать и поддерживать любые проекты. Для хранения данных используется база данных PostgreSQL. Также в проекте необходимо использовать следующие технологии:

– язык программирования Python 3.7.3;

– django-filter 2.0.0;

– django-service-objects 0.4.0;

– djangorestframework 3.8.2;

– psycopg 2 2.7.5;

– flake8 3.5.0;

– python-dotenv 0.8.2;

– django-cors-headers 2.4.0;

– PyJWT 1.6.4;

– requests 2.19.1.

4.3 Регистрация и авторизация

У веб-сервера существует только один пользователь – это администратор.

Для наглядности представления объектов, существуют различные UML-диаграммы. Одна из представленных ниже диаграмм называется диаграмма использования. «Вариант использования (use case) – это описание множества последовательных действий, включая их варианты, выполняемых системой с целью получения значимого результата для действующего лица изображается в виде эллипса». [2, 240] Общая диаграмма вариантов использования программной платформы представлена на рисунке 4.1.

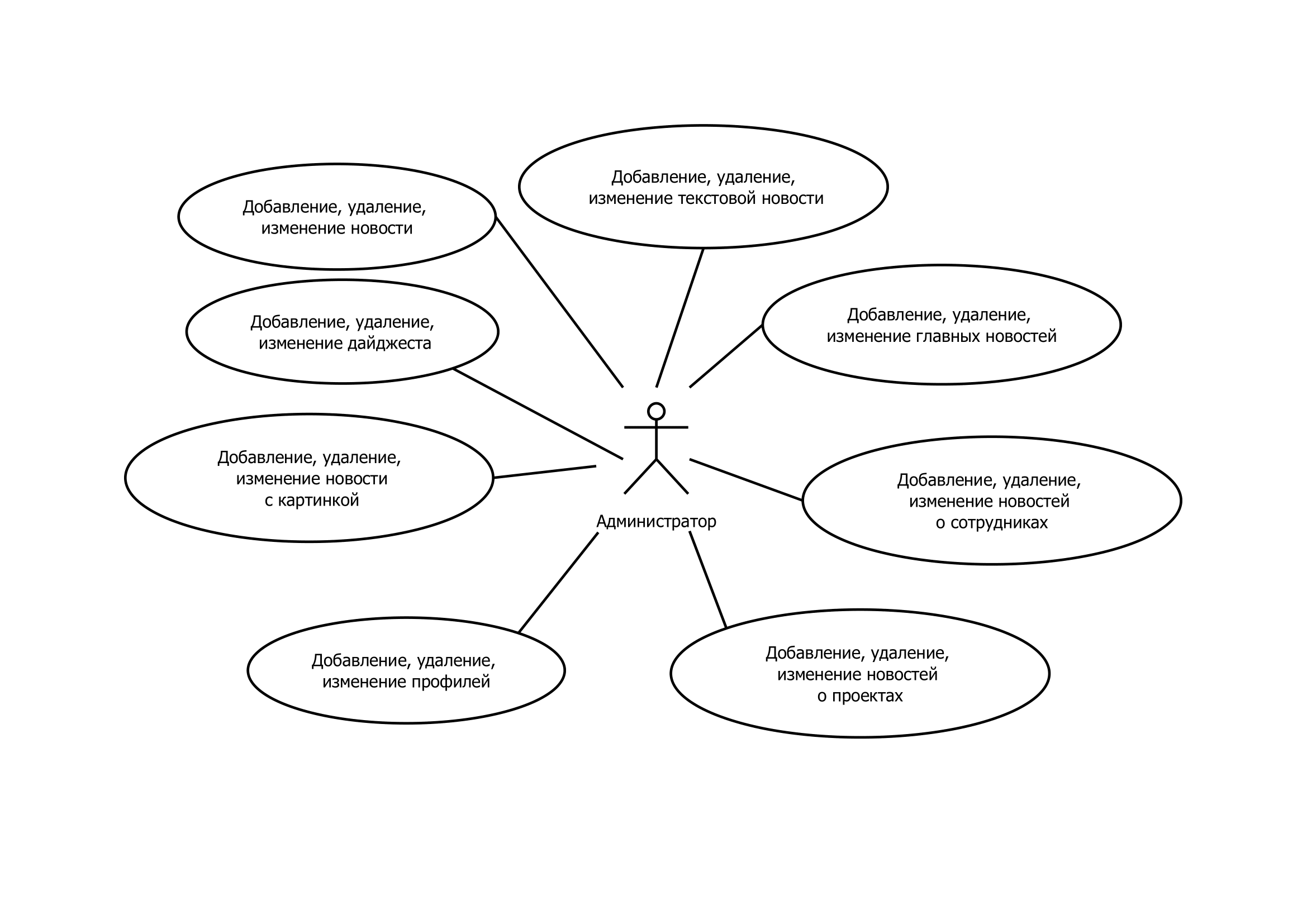


Рисунок 4.1 – Диаграмма вариантов использования

Регистрация Администратора должна происходить в тот момент, когда разработчик серверной части создаёт администрирование и базу данных. Он создаёт учётную запись для входа. Для авторизации Администратор должен запустить веб-сервер, затем перейти в браузере либо локально, либо глобально в администрирование и ввести следующие данные:

– логин (строковый тип, до 100 символов);

– пароль (не менее 9 символов, до 50 символов).

Сайт не предусматривает возможности регистрации для сотрудников. Данные для входа в учетную запись (логин и пароль) сотрудники предприятия получают от администратора.

4.4 Дайджесты

Раздел «Дайджесты» необходим для создания дайджеста и последующим её распределением по разделам, группам. Чтобы добавить новый дайджест, администратору необходимо нажать на «Digests», затем нажать на «Добавить digest». Открывается интерфейс для добавления дайджеста и предоставляются следующие поля:

– заголовок (символьный тип, не более 128 символов);

– дата (в формате DD.MM.YYYY, где В – день, M - месяц, Y – год);

– публикация (логический тип).

Здесь же можно создать ещё один пункт для создания какой-либо новости. Предоставляются следующие поля:

– заголовок;

– тип новости (новость с текстом, новость с картинкой, главные новости, новость о проекте, новость о сотруднике);

– позиция;

– важность новости.

4.5 Новости

Раздел «Новости» необходим для создания новости, в которой могут содержаться различные категории из других разделов. При добавлении новой новости предоставляются следующие поля:

– дайджест (выбор добавленных дайджестов из раздела «Дайджесты»);

– заголовок (символьный тип, не более 128 символов);

– тип (символьный тип, не более 128 символов);

– позиция (целочисленный тип данных);

– важность (логический тип данных).

4.6 Новости с картинкой

Ещё один раздел новостей – новости с картинкой. В нём администратору необходимо добавить не только текст, но и фотографию. При добавлении новой новости с картинкой предоставляются следующие поля:

– новости (выбор добавленных новостей из раздела «Новости»);

– контент (текстовая информация о новости);

– фото (URL-адрес).

4.7 Текстовые новости

Текстовые новости представляют собой важную информацию в виде текста. Здесь должны присутствовать только два поля:

– новости (выбор добавленных новостей из раздела «Новости»);

– контент (текстовая информация о новости).

4.8 Главные новости

Главные новости – это раздел сайта, который должен содержать главные события, происходящие внутри предприятия. При добавлении новой главной новости предоставляются следующие поля:

– новости (выбор добавленных новостей из раздела «Новости»);

– контент (добавление текстовой информации);

– фото (URL-адрес).

4.9 Новости сотрудников

Данный раздел нужен для того, чтобы отображать изменения о статусах сотрудников, их должностях и разрядах. Предоставляются следующие поля:

– новости (выбор новости из раздела «Новости»).

4.10 Профили

В разделе «Профили» хранится необходимая информация о сотруднике предприятия. Предоставляются следующие поля:

– имя (символьный тип, до 128 символов);

– фамилия (символьный тип, до 128 символов);

– фото (URL-адрес);

– разряд (целочисленный тип);

– должность (символьный тип, до 128 символов).

4.11 Новости проектов

В компании могут разрабатываться различные проекты. Для того, чтобы сотрудники узнавали, какие проекты уже реализованы, необходимо создать раздел «Новости проектов». Поля этого раздела следующие:

– новости (выбор добавленных новостей из раздела «Новости»);

– контент (символьный тип, до 256 символов);

– фото (URL-адрес);

– Google Play (URL-адрес);

– App Store (URL-адрес);

– браузер (URL-адрес);

– сотрудник (выбор сотрудников из раздела «Профили»).

# 5 Обзор используемых технологий и сред разработки

Для разработки каждого ПО необходимо определиться, какой стек технологий будет использоваться. После проведённого анализа технического задания был выбран язык программирования высокого уровня Python, а также среда разработки JetBrains PyCharm Professional 2019. Среди различных существующих веб-фреймворков были выбраны Django Web Framework, а также Django Rest Framework. Для физического проектирования БД для проекта была использована PostgreSQL.

5.1 Язык программирования Python

Язык программирования высокого уровня Python – достаточно простой и мощный для изучения ЯП. В нём используются эффективные высокоуровневые структуры данных. Он предлагает простой, но эффективный подход к объектно-ориентированному программированию. Этот язык:

– имеет множество веб-фреймворков;

– имеет простой синтаксис;

– перспективен на рынке труда;

– сочетает высокоуровневые и низкоуровневые концепции, обеспечивая для них удобные интерфейсы.

Интерпретатор Python легко расширяется с помощью различных библиотек, написанных на C и C++. Python можно применять как язык расширений для настраиваемых приложений. Также его можно использовать в качестве скриптов разных областях программирования. Возможности и особенности ЯП Python:

– классы являются одновременно и объектами со всеми ниже приведёнными возможностями;

– классы, вложенные в функции;

– классовые и статические методы, классовые поля;

– инкапсуляция, которая делится два уровня – общедоступные и скрытые методы и поля;

– полиморфизм и наследование;

– cкрытые члены тоже можно использовать. Только ни помечены как скрытые особыми именами;

– специальные методы, которые могут управлять жизненным циклом объектов. Это конструкторы, деструкторы и распределители памяти;

– интроспекция;

– существует возможность имитации поля с помощью функций;

– метапрограммирование. Включает в себя триггеры на создание классов, а также управление их созданием;

– вызываемые объекты, поведение которых аналогично функциям.

«В языке Python имеются интерфейсы доступа ко всем основным реляционным базам данных – Sybase, Oracle, Informix, ODBC, MySQL, PostgreSQL, SQLite и многим другим. В мире Python существует также переносимый прикладной программный интерфейс баз данных, предназначенный для доступа к базам данных SQL из сценариев на языке Python, который унифицирует доступ к различным базам данных». [3, 50]

5.2 JetBrains PyCharm Educational 2019

PyCharm – это интеллектуальная среда разработки, которая включает в большой набор средств для эффективной разработки на языке Python. Компания JetBrains выпускает три версии данного программного продукта – бесплатную версию PyCharm Community Edition, образовательную версию PyCharm Educational Editon, а также PyCharm Professional Edition, поддерживающую большой набор возможностей. Для разработки веб-сервиса DiDi использовалась образовательная версия PyCharm. IDE предоставляет удобную навигацию по коду, а также различные виды рефакторинга. PyCharm выполняет инспекцию и автодополнение во время написания кода. Она основывается на информации, которую получает во время исполнения кода. Особенности PyCharm Educational Edition:

– полнофункциональная IDE для разработки на Python, в том числе для многоязычных веб-приложений с фреймворками;

– поддержка фреймворков Django, Flask, Google App Engine, Pyramid, web2py;

– поддержка различных языков программирования и разметки, таких как JavaScript, CoffeeScript, TypeScript, HTML, CSS, Cython и другие;

– обнаружение кода, который может повторяться;

– удалённая разработка;

– поддержка работы с базами данных.

5.3 Django Web Framework

Django Web Framework находится в свободном доступе, даёт возможность заметно упростить процесс веб-разработки, так как разработчик может сфокусироваться на процессе дизайна и разработке функционала приложения. «Поскольку Django был разработан в бурлящей среде новостного агентства, он призван сделать веб-разработку простой и быстрой». [4] Таким образом, Django – это отличный инструмент для стартапов, когда веб-дизайн должен отображать концепцию и цели компании. Преимущества данного фреймворка:

1. Быстрота. Django был разработан для того, чтобы помочь разработчикам создавать приложения настолько быстро, на сколько это возможно. Этот фреймворк экономит время и ресурсы, начиная с формирования идеи, заканчивая разработкой и выпуском проекта. Его можно назвать идеальным решением для разработчиков, для которых вопрос крайних сроков выполнения задачи стоит в приоритете.

2. Полная комплектация. Django работает большим количеством разных функций и библиотек, которые заметно помогают с аутентификацией пользователя, картами сайта, администрированием содержимого, RSS и многим другим. Данные аспекты помогают осуществить каждый этап веб-разработки.

3. Безопасность. В процессе эксплуатации ПО, разработчики, администраторы и пользователи могут столкнуться разными угрозами безопасности, такими как инъекции SQL, clickjacking, кросс-сайтовый скриптинг, кросс-сайт подлоги и так далее. Django предоставляет возможность защитить проект от ошибок, связанных с безопасностью. Также фреймворк предоставляет систему логинов и паролей пользовательской аутентификации для наиболее эффективного использования администрирования Django.

4. Масштабируемость. Django Web Framework наилучшим образом подходит для работы с самыми высокими трафиками. Следовательно, большое количество загруженных сайтов используют Django для удовлетворения требований, связанных с трафиком.

5. Разносторонность. Менеджмент контента, научные вычислительные платформы, даже крупные организации – со всем этим можно эффективно справляться при помощи Django.

5.4 Django Rest Framework

«Django REST Framework – это мощный и гибкий инструментарий для создания веб-API.

Некоторые причины, по которым вы можете захотеть использовать каркас REST:

– API для просмотра веб-страниц – это огромный успех в удобстве использования для разработчиков;

– политики аутентификации, включая пакеты для OAuth1a и OAuth2;

– сериализация, которая поддерживает источники данных ORM и не-ORM». [5]

DRF можно гибко настраивать. Имеется возможность использовать функциональные представления, если разработчик не нуждается в более мощные функциях. Также для DRF предоставляется большая документация и отличная поддержка сообщества. Данный микрофреймворк пользуется доверием всемирно признанных компаний, таких как Mozilla, Red Hat, Heroku и Eventbrite.

5.5 PostgreSQL

PostgreSQL – это объектно-реляционная система управления базами данных, которая является open-source продуктом, то есть распространяется свободно без какой-либо дополнительной платы. Это наиболее развитая из открытых СУБД в мире и являющаяся реальной альтернативой коммерческим базам данных.

Функции исполняются на сервере, а не на клиенте БД, а также они являются блоками кода и могут быть написаны с использованием различных языков программирования:

– встроенный процедурный язык PL/pgSQL, во многом аналогичный языку PL/SQL, используемому в СУБД Oracle;

– Скриптовые языки Lua, Perl, PHP, Python, Ruby, Scheme, Javascript;

– Классические языки C, C++, Java;

– Статистический язык R.

PostgreSQL допускает использование функций, возвращающих набор записей, который далее можно использовать так же, как и результат выполнения обычного запроса. Функции могут выполняться как с правами их создателя, так и с правами текущего пользователя. Иногда функции отождествляются с хранимыми процедурами, однако между этими понятиями есть различие. Существует возможность написать автономные блоки кода, которые позволяют выполнять их на процедурных языках без написания функций, непосредственно в клиенте.

В самом начале PostgreSQL использовалась для исследования алгоритмов баз данных в академических проектах, а также в различных университетах. Сейчас же её область применения значительно расширилась. В настоящее время БД PostgreSQL полностью обслуживает различные доменные зоны, например, .org, .info и многотерабайтные хранилища данных, такие как BASF и Lycos. Из российских проектов, использующих PostgreSQL, наиболее известными является портал Рамблер, а также федеральные порталы Минобразования.

# 6 Проектирование базы данных

Разработка программной платформы новостных дайджестов – достаточно кропотливый процесс, который разделяется на разные этапы проектирования, реализации и эксплуатации. Начальный этап разработки платформы заключается в проектировании концептуальной модели БД. Этот этап подразумевает изучение анализа предметной области, а также технического задания, на основе которых определяются объекты моделирования, связи между ними и их свойства. Следующий этап – проектирование БД на логическом уровне. На данном этапе необходимо определить, какая система управления баз данных будет использована. Затем концептуальная модель переводится в логическую. Последний этап заключается в проектировании физической модели. Производится создание объектов в СУБД, заполнение данными, построение структур и написание запросов.

6.1 Проектирование концептуальной модели базы данных

Начальная фаза разработки программной платформы начинается с анализа предметной области и технического задания. На их основе в соответствии с реализованными вариантами использования выделяются модели представлений данных. Проанализировав поставленную задачу, были созданы следующие объекты: «Дайджесты», «Новости», «Новости с изображением», «Новости с текстом», «Большие новости», «Новости о сотрудниках», «Профили», «Карточки сотрудников», «Новости о проектах», «Пользователи», «Пользователи дайджестов», «Фавориты» и «Группы».

Между объектами заданы связи, благодаря которым они объединены в единую модель. Сущности «Дайджесты» и «Новости», «Новости о сотрудниках» и «Карточки сотрудников», «Пользователи» и «Пользователи дайджестов», «Дайджесты» и «Пользователи дайджестов», «Пользователи» и «Фавориты», а также «Новости» и «Фавориты» связаны отношением «один ко многим». Сущности «Новости с изображением» и «Новости», «Новости с текстом» и «Новости», «Большие новости» и «Новости», «Новости о сотрудниках» и «Новости», а также «Новости о проектах» и «Новости» связаны отношением «один к одному». Между сущностями «Новости о проектах» и «Профили» задана связь «многие ко многим».

Концептуальная модель данных программной платформы представлена на рисунке 6.1.

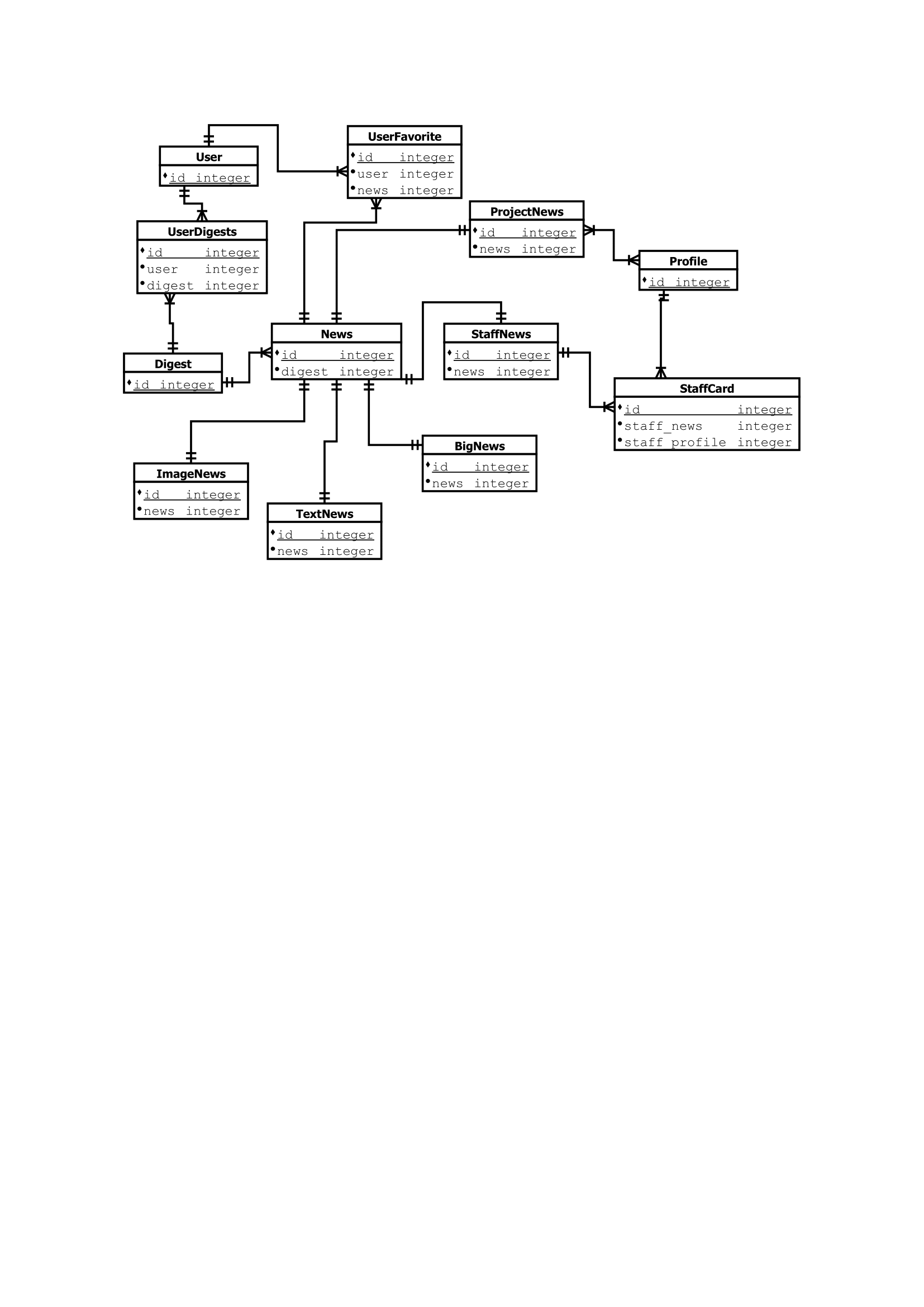


Рисунок 6.1 – Концептуальная модель базы данных программной платформы новостных дайджестов

6.2 Проектирование логической модели базы данных

Следующим шагом в проектировании программной платформы новостных дайджестов является упрощение концептуальной модели путём создания смежных таблиц между таблицами со связью «многие ко многим», так как в реляционной модели базы данных (РБД) эта связь не может быть представлена прямым образом. Между таблицами «Новости о проектах» и «Профили» установлена связь «многие ко многим». Для упрощения связи была создана таблица «Профили новостей о проектах».

Логическая модель базы данных программной платформы новостных дайджестов представлена на рисунке 6.2.

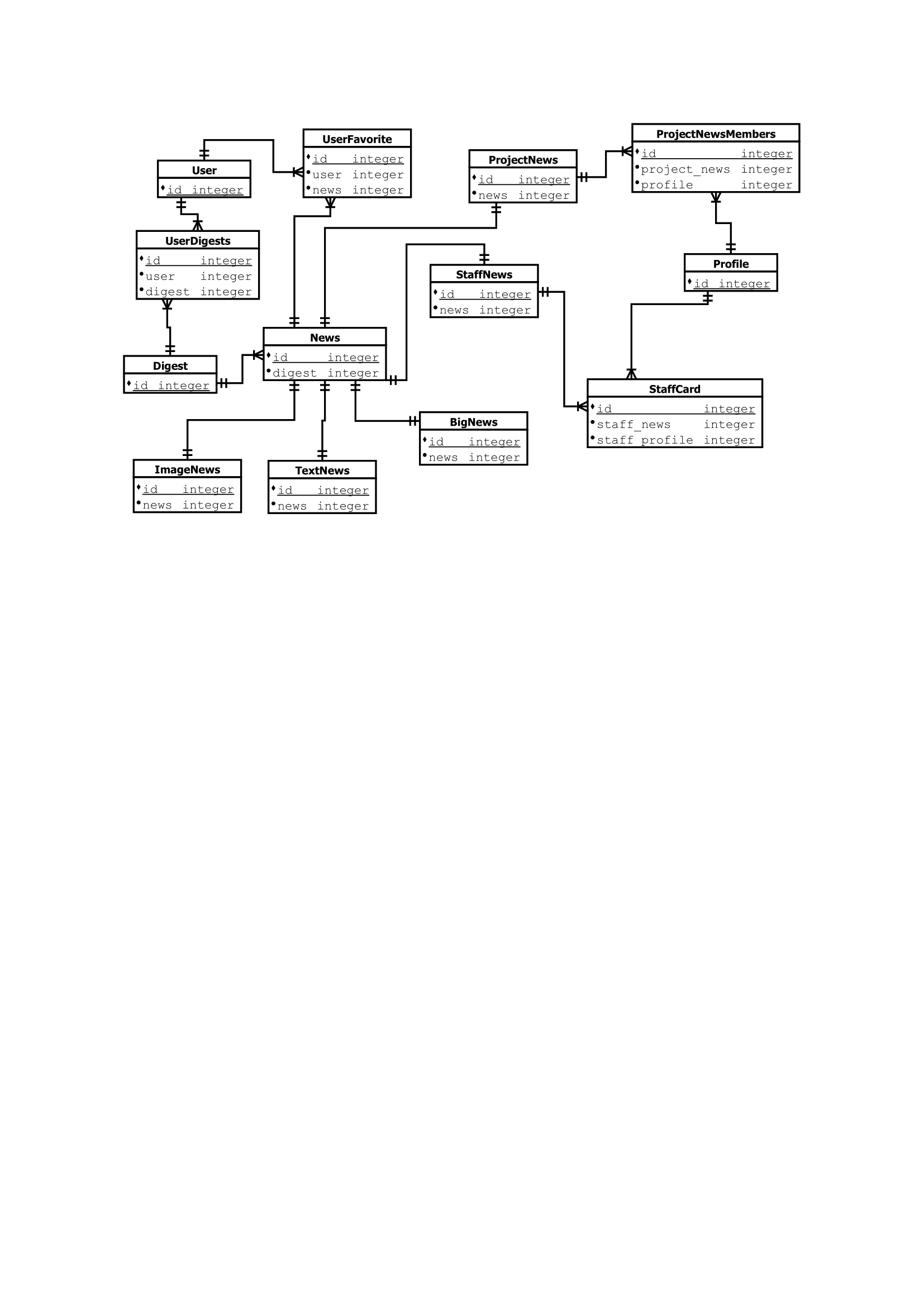


Рисунок 6.2 – Логическая модель базы данных программной платформы новостных дайджестов

6.3 Проектирование реляционной модели базы данных

Ещё один этап разработки – проектирование реляционной модели БД. Реляционная модель данных– это совокупность данных, состоящая из набора двумерных таблиц. Главными характеристиками РМД являются:

– удобное представление таблиц для пользователя;

– простота структуры данных;

– использование формального аппарата алгебры отношений для обработки данных;

– использование реляционного исчисления для обработки данных.

В РМД двумерные таблицы представлены в виде двумерных массивов. В свою очередь двумерные массивы имеют следующие свойства:

– каждая ячейка таблицы – это один элемент данных;

– все столбцы в таблице однородные, т.е. все элементы в столбце имеют одинаковый тип данных и одинаковую длину;

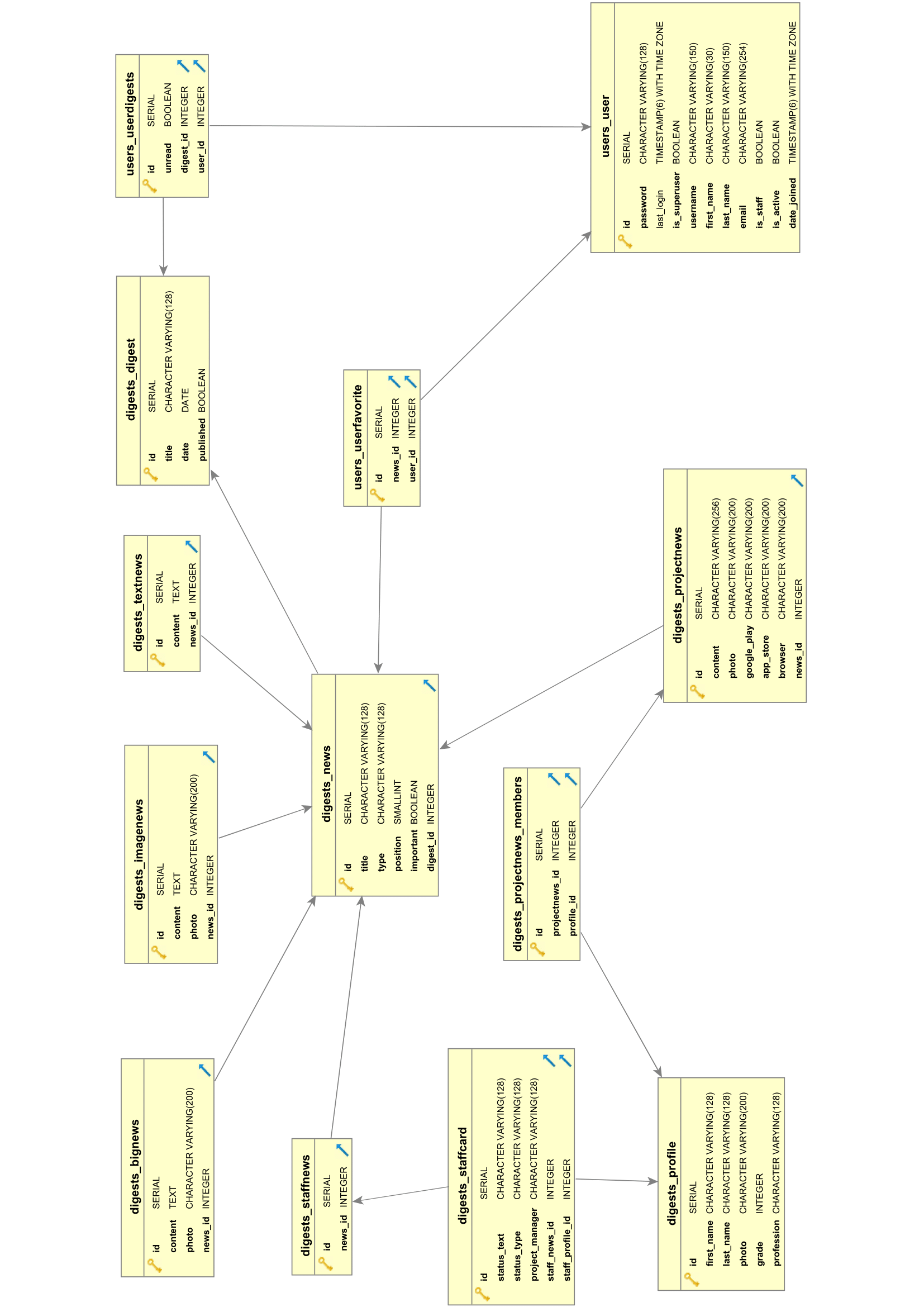
– каждый столбец имеет уникальное имя;

– одинаковые строки в таблице отсутствуют;

– порядок следования строк и столбцов может быть произвольным.

После проведённого анализа был сделан вывод, что БД больше в нормализации не нуждается. В РМД были добавлены атрибуты и их типы для дальнейшего физического проектирования БД в PostgreSQL. В итоге получилась следующая реляционная модель (рисунок 6.3).

Рисунок 6.3 – Реляционная модель базы данных программной платформы новостных дайджестов



6.4 Физическое проектирование базы данных

Благодаря связке Django Web Framework и PostgreSQL можно не работать с БД напрямую, а создавать модели в проекте приложения, описывая их кодом Python. Для реализации проекта нужно разделить модели на три группы:

– Users;

– Digests;

– Users and groups.

В ходе реализации были созданы следующие модели, представленные в таблице 6.1:

Таблица 6.1 – Группы и модели

|  |  |
| --- | --- |
| **Группы** | **Модели** |
| Users | User digests |
| User favorites |
| Users |
| Digests | Big news |
| Digests |
| Image news |
| News |
| Profiles |
| Project news |
| Staff cards |
| Staff news |
| Text news |
| Users and groups | Groups |

«Любая таблица представляет собой именованный набор строк. Все строки таблицы имеют одинаковый набор именованных столбцов, при этом каждому столбцу назначается определённый тип данных. Хотя порядок столбцов во всех строках фиксирован, важно помнить, что SQL не гарантирует какой-либо порядок строк в таблице (хотя их можно явно отсортировать при выводе).

Таблицы объединяются в базы данных, а набор баз данных, управляемый одним экземпляром сервера PostgreSQL, образует кластер баз данных». [6] Пример модели «Digests» приведён в листинге 6.1:

Листинг 6.1 – Модель «Digests»

class Digest(models.Model):

title = models.CharField(max\_length=128)

date = models.DateField(default=date.today)

published = models.BooleanField(default=False)

objects = DigestsQueryset.as\_manager()

def \_\_str\_\_(self):

return f'{self.title}'

Модель «Digests» задаётся классом «Digests». В классе описаны атрибуты и типы данных. Атрибуту «title» присваивается строковый тип «CharField» с максимальной длиной строки 128 символов. Атрибуту «date» присваивается тип с датой «DateField» с датой, которая была во время добавления данных по умолчанию. Полю «published» присваивается значение «BooleanField», тем самым означая, что данное поле имеет логический тип данных со значением «False» по умолчанию. Поле «objects» имеет значение «DigestsQueryset.as\_manager()». В классе «DigestsQueryset» реализована логика использования транзакций для данной модели. Функция «\_\_str\_\_(self):» возвращает значение «f'{self.title}'». Это необходимо для отображения объектов в интерфейсе администратора Django и в качестве значения, вставляемого в шаблон, при отображении объекта.

После описания моделей необходимо добавить их в администрирование Django путём импортирования в файл admin.py и вызовов определённых функций.

Затем на основе этих моделей создаются миграции в БД. В результате этих операций формируется база данных PostgreSQL с необходимыми сущностями. Чтобы создать новые миграции, в терминале среды разработки PyCharm необходимо выполнить следующую команду:

python manage.py makemigrations

Затем требуется применить созданные миграции. Для выполняется следующая команда:

python manage.py migrate

После выполнения вышеописанных операций создаётся администрирование Django, где можно добавлять, изменять и удалять нужные данные. Для запуска веб-сервера администрирования Django необходимо создать суперпользователя. Для этого в терминале PyCharm вводится команда:

python manage.py createsuperuser

Затем в терминале вводится логин, пароль и адрес электронной почты. После этого нужно запустить сервер командой:

python manage.py runserver

Далее для локального доступа строке браузера необходимо перейти по адресу http://127.0.0.1:8000/admin/.

Для того, чтобы создать глобальный адрес, нужно скачать специальную консольную утилиту ngrok, запустить командную строку от имени администратора и ввести команду:

ngrok http 8000

После выполнения этой команды в командной строке запускается данный сервис. В интерфейсе окна командной строки генерируется и отображается глобальный адрес сайта, по которому любой пользователь может зайти на сайт (рисунок 6.4).

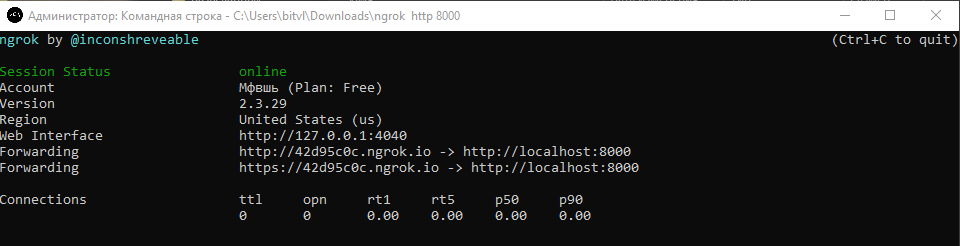


Рисунок 6.4 – Работа ngrok

Программа ngrok каждый раз после запуска генерирует новый URL-адрес. Создаются два вида адресов: URL с незащищённым HTTP протоколом и защищённым HTTPS протоколом. Также перейдя по адресу, находящемуся в поле «Web Interface», в браузере можно открыть историю запросов к веб-серверу. Перейдя по URL-адресам, назначенным в полях «Forwarding», в браузере представляется окно входа в администрирование Django, где необходимо ввести логин и пароль суперпользователя (рисунок 6.5).

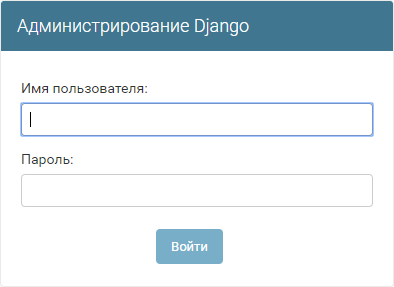


Рисунок 6.5 – Окно входа в администрирование Django

После ввода логина и пароля совершается вход в администрирование Django, где можно посмотреть все созданные группы и модели (рисунок 6.6).

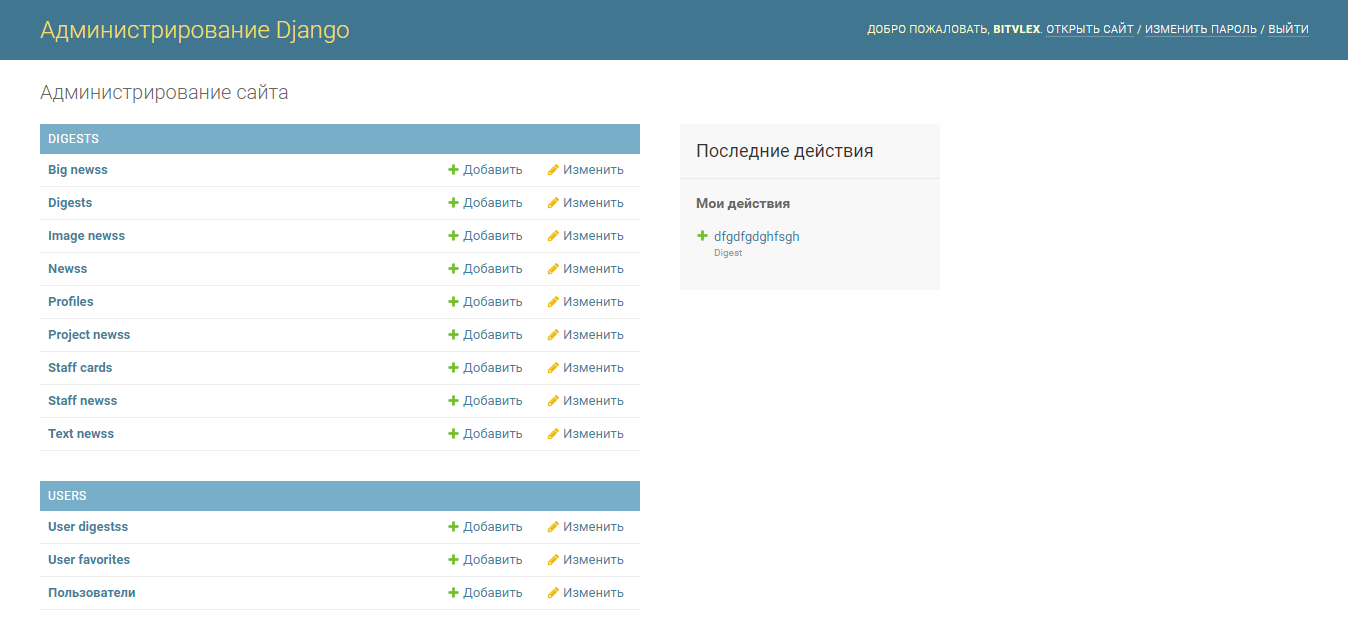


Рисунок 6.6 – Администрирование Django

При нажатии на какую-либо модель, например, «Digests», открывается интерфейс, где предоставляется возможность выбора из списка уже созданных дайджестов для последующего их изменения или удаления, а также для создания новых (рисунок 6.7).

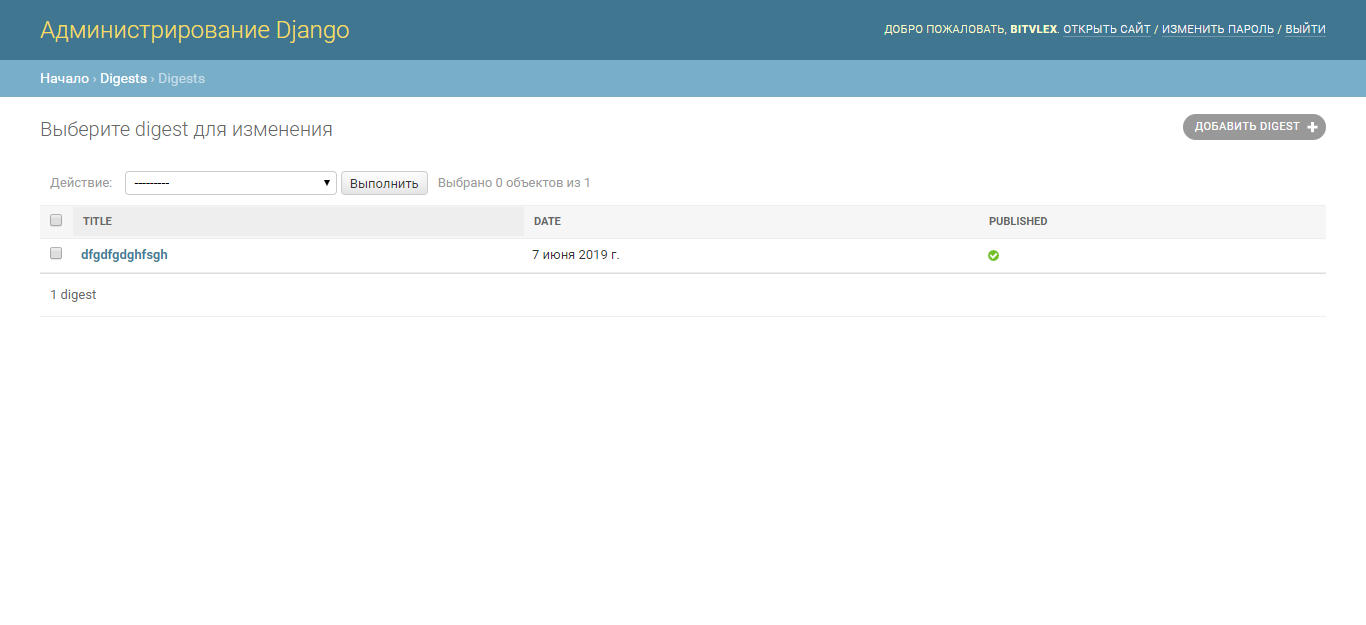


Рисунок 6.7 – Редактирование объектов в администрировании Django

# 7 Реализация работы сервера с API

После создания моделей и их миграций в базу данных PostgreSQL необходимо создать API для взаимодействия с клиентской частью проекта. API – это программный интерфейс приложений и набор различных средств, который позволяет одной программе взаимодействовать с другой. В программной платформе новостных дайджестов работа с API реализована через библиотеку, которая работает со стандартными моделями Django Django Rest Framework (DRF). С помощью этого фреймворка можно реализовать гибкий и мощный API для проекта.

Реализация работы с API делится на несколько этапов. Для начала необходимо создать сериализаторы для преобразования данных в формат JSON и последующей их передачи через API. Следующим шагом нужно реализовать виды, в которых будут определены доступные операции. Также с помощью маршрутизаторов необходимо организовать передачу видов через API.

7.1 Сериализация

Модели Django интуитивно представляют данные, хранящиеся в базе, но API должен передавать информацию в менее сложной структуре. Хотя данные будут представлены как экземпляры классов Model, их необходимо перевести в формат JSON для передачи через API.

«Сериализатор DRF производит это преобразование. Когда пользователь передает информацию (например, создание нового экземпляра) через API, сериализатор берет данные, проверяет их и преобразует в нечто, что Django может сложить в экземпляр модели. Аналогичным образом, когда пользователь обращается к информации через API, соответствующие экземпляры передаются в сериализатор, который преобразовывает их в формат, который может быть легко передан пользователю как JSON». [7]

Наиболее распространенной формой, которую принимает сериализатор DRF, является та, который привязана непосредственно к модели Django.

Настройки fields позволяют точно указать, какие поля доступны этому сериализатору. В качестве альтернативы, может быть установлен exclude вместо fields, которое будет включать все поля модели, кроме тех, которые указаны в exclude.

Сериализаторы – это невероятно гибкий и мощный компонент DRF. Хотя подключение сериализатора к модели является наиболее распространенным, сериализаторы могут использоваться для создания любой структуры данных Python через API в соответствии с определенными параметрами. Пример сериализатора приведён в листинге 7.1.

Листинг 7.1 – Сериализаторы для модели «Digests»

class DigestCreateUpdateDeleteSerializer(serializers.ModelSerializer):

news = NewsSerializer(many=True, read\_only=True)

class Meta:

model = Digest

fields = ('id', 'title', 'date', 'news')

extra\_kwargs = {

'date': {

'required': True

},

}

class DigestReadSerializer(serializers.ModelSerializer):

news = NewsSerializer(many=True, read\_only=True)

unread = serializers.SerializerMethodField()

class Meta:

model = Digest

fields = ('id', 'title', 'date', 'news',

'unread')

extra\_kwargs = {

'date': {

'required': True

},

}

7.2 Представления

Сериализатор анализирует информацию в обоих направлениях (чтение и запись), тогда как представление - это тот код, в котором определены доступные операции. Наиболее распространенным представлением является ModelViewSet, который имеет следующие встроенные операции:

1. Создание экземпляра: create().

2. Получение и чтение экземпляра: retrieve().

3. Обновление экземпляра (все или только выбранные поля): update() или partial\_update().

4. Уничтожение и удаление экземпляра: destroy().

5. Список экземпляров (с разбивкой по страницам по умолчанию): list().

Каждая из этих функций может быть переопределена, если требуется другое поведение, но стандартная функциональность работает с минимальным кодом.

Если необходимы дополнительные настройки, можно использовать общие представления, вместо ModelViewSet или даже отдельные пользовательские представления. Пример представления приведён в листинге 7.2.

Листинг 7.2 – Представление для модели «Digests»

class DigestsViewset(viewsets.ModelViewSet):

queryset = Digest.objects.all()

serializer\_class = DigestReadSerializer

filterset\_class = DigestFilter

permission\_classes = (IsAdminOrReadOnly,)

def get\_queryset(self):

queryset = super().get\_queryset()

queryset = (

queryset

.prefetch\_news\_with\_favorite(user=self.request.user)

.get\_with\_unread\_status(user=self.request.user)

.order\_by('-date')

.distinct()

)

if not self.request.user.is\_staff:

return queryset.get\_published()

return queryset

def \_is\_filtering(self):

return any([

self.request.query\_params.get('important'),

self.request.query\_params.get('favorite'),

self.request.query\_params.get('search'),

])

7.3 Маршрутизация

И наконец, маршрутизаторы: они предоставляют верхний уровень API. Чтобы избежать создания бесконечных URL-адресов вида: «списки», «детали» и «изменить», маршрутизаторы DRF объединяют все URL-адреса, необходимые для данного вида в одну строку для каждого представления. Затем все представления, которые зарегистрированы в маршрутизаторе, можно добавить к обычным url\_patterns.

Теперь через API можно получить данные точно также, как и любые другие обычные страницы Django. Пример маршрутизатора приведён в листинге 7.3.

Листинг 7.3 – Маршрутизатор для модели «Digests»

from rest\_framework import routers

from . import views

router = routers.SimpleRouter()

router.register(r'digests', views.DigestsViewset, base\_name='digests')

router.register(r'news', views.NewsViewset, base\_name='news')

urlpatterns = router.urls

# 8 Тестирование программной платформы

Один из важных этапов в разработке программного обеспечения заключается в тестировании. Необходимо проверить поведение программы при различных ситуациях. Также нужно знать, как будет работать программа на выходе при различных входных тестовых данных. Django Rest Framework предлагает отличный набор инструментов для написания тестов. Набор тестовых данных был сформирован таким образом, чтобы покрыть все возможные ветви методов проекта. Программная платформа была протестирована добавление, удаление, хранение объектов и данных. Также происходит проверка на непрочитанные дайджесты, добавление и удаление новостей, дайджестов, а также получение списка всех дайджестов. Для проверки взаимодействия пользователей и дайджестов также были написаны тестовые наборы. Для удобства все тестовые функции занесены в таблицу 8.1.

Таблица 8.1 – Тестовые функции

|  |  |
| --- | --- |
| **Название тестовой функции** | **Описание** |
| 1. test\_unread\_digests | Проверка на непрочитанные дайджесты, логический флаг в списке дайджестов, а также тестирование удаления из непрочитанных дайджестов |
| 2. test\_create\_digest | Тестирование на добавление дайджеста от имени администратора |
| 3. test\_get\_digest\_list | Тест на получение списка дайджестов |
| 4. test\_get\_digest\_not\_found | Тест на «дайджест не найден» |
| 5. test\_delete\_digests | Удаление дайджестов от имени администратора |
| 6. test\_filter\_important | Фильтр важности |

*Окончание таблицы 8.1*

|  |  |
| --- | --- |
| 7. test\_filter\_dates | Фильтр дат |
| 8. test\_get\_dates | Получение даты |
| 9. test\_delete\_news | Удаление новости |
| 10. test\_patch\_news | Комплект новостей |
| 11. test\_change\_type\_forbidden | Тест на запрет изменения типа |
| 12. test\_create\_favorite | Добавить фаворитов |
| 13. test\_get\_favorites | Получить фаворитов |
| 14. test\_delete\_favorite | Удалить фаворитов |
| 15. test\_get\_filtered\_favorite\_news | Фильтр новостей фаворитов |

Пример тестовой функции приведён в листинге 8.1.

Листинг 8.1 – Тестовая функция «test\_unread\_digests»

def test\_unread\_digests(self):

digests\_count = Digest.objects.all().count()

url = reverse('digests:digests-unread')

response = self.client.get(url)

self.assertEqual(response.status\_code, status.HTTP\_200\_OK, response.data)

self.assertEqual(response.data.get('count'), digests\_count, response.data)

url = reverse('digests:digests-list')

response = self.client.get(url)

self.assertEqual(response.status\_code, status.HTTP\_200\_OK, response.data)

self.assertTrue(response.data['results'][0].get('unread'))

self.assertTrue(response.data['results'][1].get('unread'))

self.assertTrue(response.data['results'][2].get('unread'))

unread\_digest\_id = response.data['results'][0].get('id')

url = reverse('digests:digests-detail', args=[unread\_digest\_id])

response = self.client.get(url)

self.assertEqual(response.status\_code, status.HTTP\_200\_OK, response.data)

url = reverse('digests:digests-list')

response = self.client.get(url)

self.assertEqual(response.status\_code, status.HTTP\_200\_OK, response.data)

self.assertFalse(response.data['results'][0].get('unread'))

После написания тестовых наборов необходимо их запустить. Для этого в терминале PyCharm нужно выполнить команду:

python manage.py test

В терминале PyCharm видно, что все тесты успешно выполнены (рисунок 8.1).

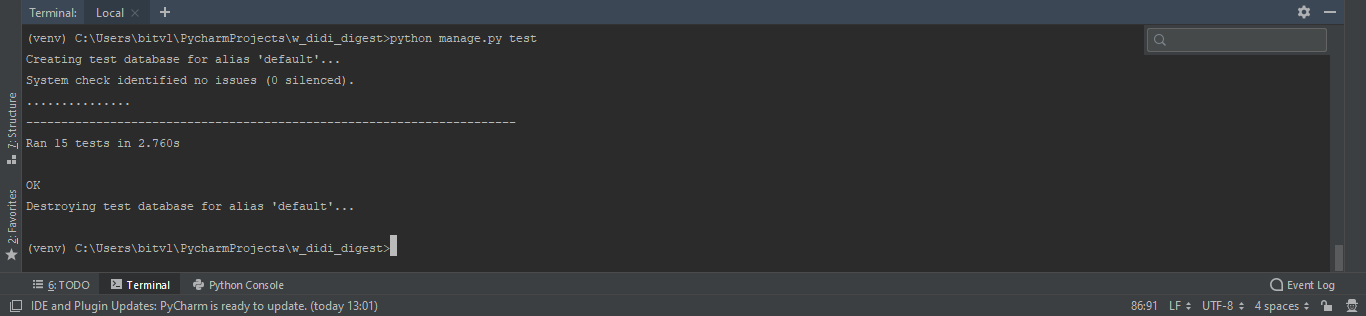


Рисунок 8.1 – Выполнение тестов

# 9 Руководство пользователя

Для просмотра домашней страницы веб-сервиса DiDi пользователю необходимо перейти на главный домен приложения (рисунок 9.1).



Рисунок 9.1 – Домашняя страница

Каждому пользователю предоставляется набор возможностей:

– просмотр всех дайджестов;

– просмотр дайджестов за выбранный месяц и год;

– просмотр выбранного дайджеста;

– просмотр важных новостей при отображении полной версии новости;

– просмотр важных новостей при отображении заголовков на домашней странице.

Только администратор веб-сервиса может отмечать и самостоятельно устанавливать:

– важность новости;

– просмотр выбранной новости по ссылке с хэшем;

– добавление новостей в избранное;

– поиск по новостям.

На главной странице веб-сервиса отображаются дайджесты с заголовками новостей. Для того, чтобы определить важность новости, слева от её названия отображается иконка звезды (рисунок 9.2).

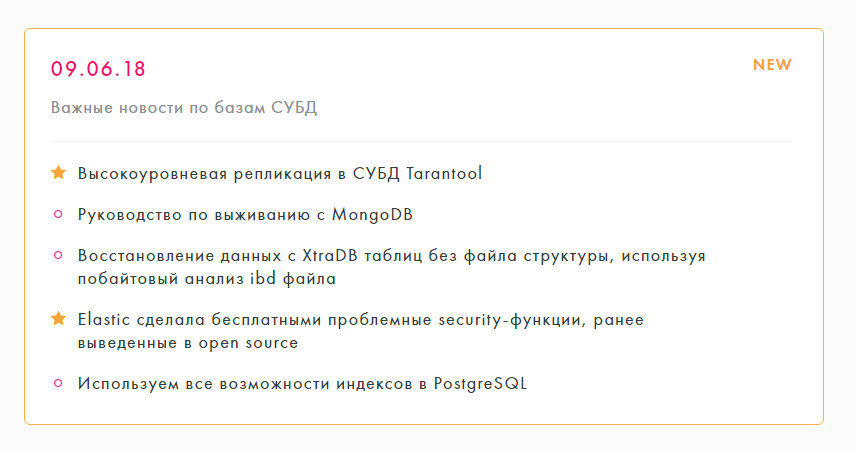


Рисунок 9.2 – Отображение важных новостей

Если по каким-либо причинам у пользователя не было возможности заходить на сайт, то при следующем его визите все непрочитанные новости будут выделены специальным контуром и отмечены, как новые, с помощью слова «NEW» в правом верхнем углу дайджеста, справа от пункта «Все новости», где находится боковая панель в будет отображено количество непрочитанных новостей (рисунок 9.3).

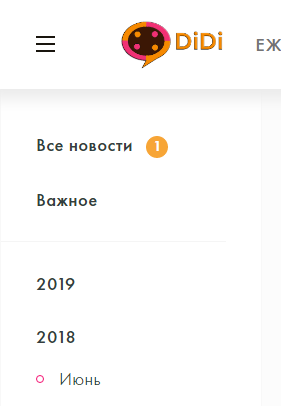


Рисунок 9.3 – Информация о количестве непрочитанных новостей в боковой панели

В программной платформе публикации новостных дайджестов пользователю предоставляется возможность добавить нужную новость в раздел «Избранное». Добавление новости в избранное показано на рисунке 9.4.

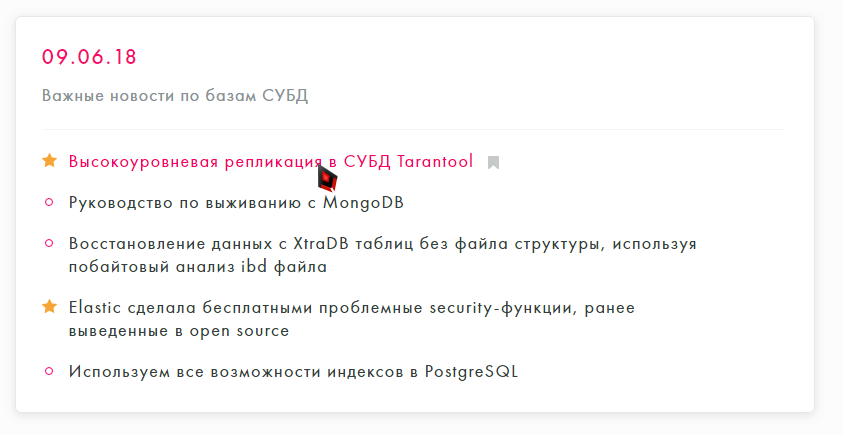


Рисунок 9.4 – Отображение иконки для добавления новости в «Избранное»

Если навести курсор мыши на заголовок новости в дайджесте, то слева от заголовка появляется иконка флага. При нажатии на эту иконку, новость отмечается как избранная. Избранные новости можно просматривать. Для этого необходимо открыть боковую панель и перейти во вкладку «Избранное» (рисунок 9.5).

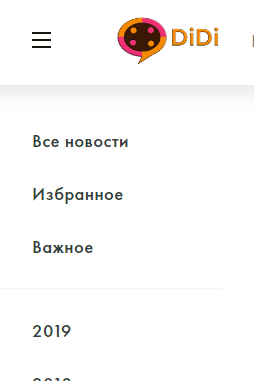


Рисунок 9.5 – Отображение пункта «Избранное» в боковой панели

Пользователю также предоставляется возможность добавлять новости в раздел «Избранное» при просмотре полной версии дайджеста. Если навести курсор мыши на заголовок новости, то слева от заголовка появляется уже известная иконка флага. Нажав на эту иконку, она окрасится в другой цвет, новость добавится в раздел «Избранное» и будет доступна для последующего просмотра (рисунок 9.6).

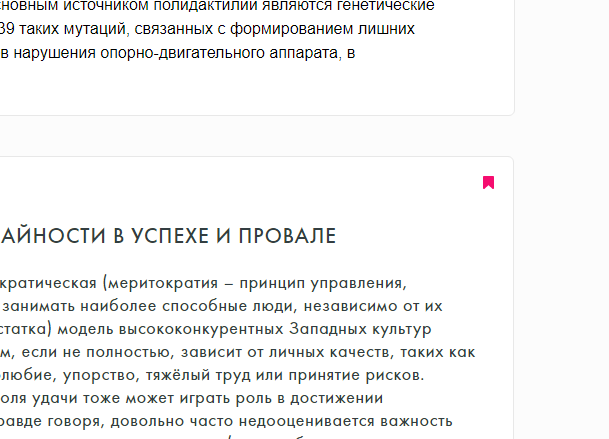


Рисунок 9.6 – Отображение новости в разделе «Избранное»

Если пользователя появится желание поделиться новостью со своими сотрудниками, он может отправить им ссылку на конкретную новость. Чтобы получить прямую ссылку на новость, необходимо нажать правой кнопкой мыши на заголовке новости. После этого нужно скопировать ссылку, либо нажать на заголовок. Далее откроется страница с полным содержимым дайджеста. При этом браузер перейдёт по странице в нужное место, где находится ссылка скопированной новости и перед пользователем откроется просмотр скопированной новости. Пример прямой ссылки для открытия необходимой новости показан на рисунке 9.7.

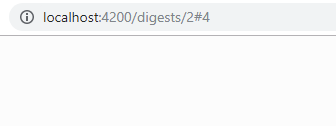


Рисунок 9.7 – URL-адрес для открытия определенной новости

# 10 Безопасность и экологичность проекта

10.1 Значение и задачи безопасности жизнедеятельности

В настоящее время всемирная сеть пользуется большой популярностью. Множество успешных компаний очень активно используют сеть Интернет для обмена различной внутрикорпоративной информацией. Это даёт высокий социально-экономический эффект, в частности, увеличивается скорость обмена информацией и повышается рост производительности труда.

Персональные электронно-вычислительные машины негативно воздействуют на человека незаметным образом. При постоянной и продолжительной работе с компьютером определённые факторы могут неблагоприятно повлиять на его самочувствие, безопасность и здоровье. Длительное времяпрепровождение за ПЭВМ может способствовать быстрому переутомлению, нарушению функций зрения, а также эмоционально-нервному перенапряжению.

Главными задачами безопасности жизнедеятельности и охраны труда является ликвидация, минимизация и снижение уровня влияния неблагоприятных факторов на человеческий организм и защита от отрицательных последствий их влияния. Для того, чтобы избежать эти негативные последствия, выполнены необходимые меры безопасности при работе с ПЭВМ, которые регламентированы в следующих нормативных документах:

– СанПиН 2.2.2/2.4.1340‑03 «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы»;

– «Трудовой кодекс Российской Федерации» от 30.12.2001 № 197-ФЗ (ред. от 10.01.2016);

– ТОИ Р‑45‑084‑01 «Типовая инструкция по охране труда при работе на персональном компьютере»;

– ГОСТ 12.0.003‑2015 «Система стандартов безопасности труда (ССБТ). Опасные и вредные производственные факторы. Классификация».

10.2 Анализ условий труда и мероприятия по защите от воздействия вредных производственных факторов

«Программная платформа новостных дайджестов» - это веб-сайт, который успешно информирует сотрудников предприятия о новых значимых событиях, происходящих внутри компании. Работа администратора сайта требует определённых усилий и концентрации внимания. Из-за этого может случится переутомление, усталость, раздражённость и нарушение функций зрения. Отрицательное воздействие на человека вычислительной техники менее выражено, сглажено многими положительными моментами. Однако у людей длительно использующих ПЭВМ могут быть отмечены такие реакции как переутомление, усталость, раздражённость и нарушение функций зрения

Чтобы минимизировать и ликвидировать последствия, описанные выше, созданы комфортные условия для работы, а также соблюдены все нормы по устранению тяжести и напряжённости трудового процесса.

Оценим напряжённость работы администратора сайта при работе с администрированием Django и БД. Документ, по которому оценивается тяжесть – Р 2.2.2006‑05 «Руководство по гигиенической оценке факторов рабочей среды и трудового процесса. Критерии и классификация условий труда». Анализ оценки напряжённости трудового процесса показан в таблице 10.1.

Таблица 10.1 – Оценка напряженности трудового процесса

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Показатели** | **Класс условий труда** | | | | | |
| **1** | | | **2** | **3.1** | **3.2** |
| 1 Интеллектуальные нагрузки | | | | | | |
| 1.1 Содержание работы | |  | | + |  |  |
| 1.2 Восприятие информации и их оценка | |  | |  | + |  |
| 1.3 Распределение функций по степени сложности задания | |  | | + |  |  |
| 1.4 Характер выполняемой работы | |  | |  | + |  |
| 2 Сенсорные нагрузки | | | | | | |
| 2.1 Длительность сосредоточенного наблюдения (% времени смены) | |  | |  |  | + |
| 2.2 Плотность сигналов (световых, звуковых) и сообщений в среднем за 1 час работы | | + | |  |  |  |
| 2.3 Число производственных объектов одновременного наблюдения | | + | |  |  |  |
| 2.4 Размер объекта различения (при расстоянии от глаз работающего до объекта различения не более 0,5 м) в мм при длительности сосредоточенного наблюдения (% времени смены) | |  | | + |  |  |
| 2.5 Работа с оптическими приборами при длительности сосредоточения наблюдения | | + | |  |  |  |
| 2.6 Наблюдение за экранами видеотерминалов (часов в смену) | |  | |  |  | + |
| 2.7 Нагрузка на слуховой анализатор | |  | | + |  |  |
| 2.8 Нагрузка на голосовой аппарат | | + | |  |  |  |
| 3 Эмоциональные нагрузки | | | | | | |
| 3.1 Степень ответственности за результат собственной деятельности. Значимость ошибки | | |  | + |  |  |
| 3.2 Степень риска для собственной жизни | | | + |  |  |  |
| 3.3 Ответственность за безопасность других лиц | | | + |  |  |  |
| 3.4 Количество конфликтных ситуаций в течение смены – от 1 до 3. | | |  | + |  |  |

*Окончание таблицы 10.1*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 Монотонность нагрузок | | | | |
| 4.1 Число элементов, необходимых для реализации простого задания или многократно повторяющихся операций |  | + |  |  |
| 4.2 Продолжительность выполнения простых заданий или повторяющихся операций |  |  | + |  |
| 4.3 Время активных действий |  |  | + |  |
| 4.4 Монотонность производственной обстановки (время пассивного наблюдения за ходом техпроцесса в % от времени смены) | + |  |  |  |
| 5 Режим работы | | | | |
| 5.1 Фактическая продолжительность рабочего дня |  | + |  |  |
| 5.2 Сменность работы | + |  |  |  |
| 5.3 Наличие регламентированных перерывов и их продолжительность | + |  |  |  |
| Количество показателей в каждом классе | 9 | 8 | 4 | 2 |
| Общая оценка напряженности труда | 3.1 | | | |

После проведённого анализа были получены следующие результаты: В 1 класс входят 9 показателей, во 2 класс – 8 показателей, в класс 3.1 – 4 показателя, а в класс 3.2 – 2 показателя. В итоге стало понятно, что по уровню тяжести и напряжённости работу администратора можно отнести к классу 3.1, так как она достаточно напряжённая. Для того, чтобы минимизировать напряжённость, усталость глаз, мышц и в целом раздражённость, введены перерывы с определённой периодичностью. Для того, чтобы уменьшить сенсорные нагрузки 2.1 и 2.6, было установлено программное обеспечение, которое оповещает об изменении на терминале. Это даёт возможность наблюдать за терминалом с более меньшим напряжением. Благодаря этим операциям два показателя 3.2 переведены в 2 класс условий труда. Общая оценка напряженности труда составляет 2, то есть допустимый.

10.3 Обеспечение электробезопасности

Для оптимальной работы веб-сервера комната обеспечена повышенной электробезопасностью. В помещение для администратора, где размещено серверное оборудование, установлены качественные бесперебойные устройства, которые защищают сервер от перегрузок. Также проведено необходимое заземление и подключены необходимые устройства электроустановок. Периодически проводятся технико-организационные мероприятия для профилактики повреждённой изоляции, уборке помещения, проверки электрооборудования и устранения выявленных неполадок.

Для администратора веб-сервера организовано специальное удобное рабочее место и созданы комфортные условия для работы. Для этого полы обеспечены изоляцией, температура поддерживается примерно в районе 21‑22°С, установлена удобная мебель. Освещённость равномерно распределена по всему помещению, не нарушая зрение.

10.4 Пожарная безопасность

Основные документы, регламентирующие нормы пожарной безопасности в офисе, являются Федеральный закон № 123-ФЗ от 22 июля 2008 г. «Технический регламент о требованиях пожарной безопасности» и Правила противопожарного режима Российской Федерации, утвержденные постановлением Правительства РФ от 25 июля 2012 г. № 390 «О противопожарном режиме».

Любое помещение предприятия обеспечено системами пожарной безопасности. За соблюдение правил и норм пожарной безопасности отвечает руководитель компании. Главные факторы, влияющие на пожарную безопасность это расположение помещения, а также его площадь, количество размещённого электрооборудования, количество сотрудников предприятия и наличие производственных помещений.

Для того, чтобы обеспечить высокий уровень пожарной безопасности, были выполнены следующие меры:

– размещены таблички с номерами пожарных служб;

– размещены планы эвакуации;

– организованы запасные выходы для оперативной эвакуации при пожаре;

– установлены необходимые системы пожаротушения;

– установлены систем пожарной сигнализации;

– размещены знаки пожарной безопасности, таблички, запрещающие курение или указывающие на какую-либо опасность.

Все вышеописанные мероприятия, при их выполнении, уменьшают возникновение чрезвычайных ситуаций. Комплекс мероприятий направлен по обеспечению снижения напряжённости трудового процесса, обеспечению электро- и пожарной безопасности. Программная платформа отвечает всем нормам, требованиям и стандартам.

# Заключение

В результате выполнения выпускной квалификационной работы была реализована программная платформа новостных дайджестов, благодаря которой сотрудники могут просматривать новостной контент, быть в курсе событий, происходящих в компании, в которой они работают.

В ходе разработки проекта были реализованы модели, выполнены миграции в БД, созданы сериализаторы для оптимальной работы с API, а также представления и маршрутизаторы. В процессе тестирования выяснилось, что система работает правильно.

Для реализации серверной части программной платформы публикации новостных дайджестов были задействованы следующие технологии:

– язык программирования высокого уровня Python;

– веб-фреймворк Django;

– микрофреймворк Django Rest;

– СУБД PostgreSQL.

Реализована передача данных на клиентское приложение через API в формате JSON.

Главная задача ВКР успешно достигнута. Программная платформа публикации новостных дайджестов может использоваться в коммерческих целях, масштабироваться и оптимизироваться под различные компании.

# Список использованных источников

1. Вахрамеева З.В., Курбангалеева И.В. Дайджест как продукт информационной деятельности // БИБЛИОСФЕРА. – 2008. – № 2. – с. 49-53

2. Буч Г., Рамбо Д., Якобсон И. Язык UML. Руководство пользователя. 2-е изд.: Пер. с англ. Мухин Н. – М.: ДМК Пресс, 2006. – 496 с.

3. Лутц М. Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с.

4. Знакомство с Django — Документация Django 1.9 [Электронный ресурс]. URL: https://djbook.ru/rel1.9/intro/overview.html (дата обращения: 29.05.2019)

5. Home - Django REST framework [Электронный ресурс]. URL: https://www.django-rest-framework.org (дата обращения: 31.05.2019)

6. PostgreSQL : Документация: 11: 2.2. Основные понятия : Компания Postgres Professional [Электронный ресурс]. URL: https://postgrespro.ru/docs/postgresql/11/tutorial-concepts/ (дата обращения: 02.06.2019)

7. Кратко о Django Rest Framework | Все о фреймворке Django [Электронный ресурс]. URL: http://django.fun/tutorials/kratko-o-django-rest-framework/ (дата обращения: 04.06.2019)