|  |  |
| --- | --- |
| Изображение выглядит как эмблема, герб, символ, нашивка  Автоматически созданное описание | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Системы обработки информации и управления

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ***

***НА ТЕМУ:***

**Сегментация объектов на спутниковых снимках**

Студент группы ИУ5-81Б **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** П.К. Шимолина

(код группы) (подпись, дата) (инициалы и фамилия)

Научный руководитель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** А. Н. Нардид

(подпись, дата) (инициалы и фамилия)

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель от кафедры **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** В.И. Терехов

(подпись, дата) (инициалы и фамилия)

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2024 г

**АННОТАЦИЯ**

В данной работе рассматривается разработка системы семантической сегментации изображений с использованием нейронных сетей. Основной целью работы является создание модели для решения задач сегментации и интеграция её в веб-сервис.

В первой главе описана предметная область и постановка задач разработки. Приведены цели и задачи, проведено сравнение с аналогичными системами, описаны типы задач сегментации и особенности семантической сегментации. Рассмотрены основные архитектуры нейронных сетей, такие как U-Net, DeepLabV3Plus и Link-Net, и проведено их сравнение.

Далее представлен процесс обучения модели DeepLabV3Plus, включая выбор набора данных, предобработку и преобразования данных, создание модели, определение метрик и обучение модели. Разработан сервис сегментации, описаны используемые технологии, архитектура backend и frontend частей сервиса. В частности, рассмотрены модели, представления и эндпоинты backend, а также маршруты и компоненты frontend.

Работа представляет собой комплексное исследование и разработку системы сегментации изображений, что позволяет применить полученные знания и навыки в области компьютерного зрения и глубокого обучения. Результаты проекта могут быть полезны для различных областей, где требуется точное и эффективное выделение объектов на изображениях.

**СОДЕРЖАНИЕ**

[ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ 5](#_Toc167314611)

[ВВЕДЕНИЕ 7](#_Toc167314612)

[1 ПОСТАНОВКА ЗАДАЧ РАЗРАБОТКИ 8](#_Toc167314613)

[1.1 Описание предметной области 8](#_Toc167314614)

[1.2 Цели и задачи 8](#_Toc167314615)

[1.3 Сравнение с аналогами 9](#_Toc167314616)

[1.3.1 Формирование критериев оценки 9](#_Toc167314617)

[1.3.2 Варианты сервисов сегментации 10](#_Toc167314618)

[1.3.3 Выбор лучшего варианта 13](#_Toc167314619)

[1.4 Типы задач сегментации 14](#_Toc167314620)

[1.5 Семантическая сегментация 15](#_Toc167314621)

[1.6 Архитектуры нейронных сетей для решения задачи семантической сегментации 16](#_Toc167314622)

[1.6.1 U-Net 18](#_Toc167314623)

[1.6.2 DeepLabV3Plus 19](#_Toc167314624)

[1.6.3 Link-Net 20](#_Toc167314625)

[1.6.4 Сравнение архитектур 22](#_Toc167314626)

[2 ОБУЧЕНИЕ МОДЕЛИ DEEPLABV3PLUS 24](#_Toc167314627)

[2.1 Набор данных 24](#_Toc167314628)

[2.2 Предобработка и преобразования 25](#_Toc167314629)

[2.3 Модель 26](#_Toc167314630)

[2.3.1 Создание модели 26](#_Toc167314631)

[2.3.2 Метрики 27](#_Toc167314632)

[2.3.3 Обучение модели 29](#_Toc167314633)

[3 СЕРВИС СЕГМЕНТАЦИИ 32](#_Toc167314634)

[3.1 Постановка задачи 32](#_Toc167314635)

[3.2 Функциональная модель 34](#_Toc167314636)

[3.3 Диаграмма сущность-связь 39](#_Toc167314637)

[3.4 Используемые технологии 41](#_Toc167314638)

[3.5 Разработка сервиса 42](#_Toc167314639)

[3.5.1 Backend 42](#_Toc167314640)

[3.5.1.1 Модели 43](#_Toc167314641)

[3.5.1.2 Представления 45](#_Toc167314642)

[3.5.1.3 Эндпоинты 47](#_Toc167314643)

[3.5.1.4 База данных 48](#_Toc167314644)

[3.5.2 Frontend 49](#_Toc167314645)

[3.5.2.1 Маршруты 50](#_Toc167314646)

[3.5.2.2 Компоненты 51](#_Toc167314647)

[ЗАКЛЮЧЕНИЕ 57](#_Toc167314648)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 58](#_Toc167314649)

[ПРИЛОЖЕНИЕ A ТЕХНИЧЕСКОЕ ЗАДАНИЕ 60](#_Toc167314650)

[ПРИЛОЖЕНИЕ B ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ 66](#_Toc167314651)

[ПРИЛОЖЕНИЕ C РУКОВОДСТВО ОПЕРАТОРА 75](#_Toc167314652)

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

API (Application Programming Interface) — это набор правил и инструментов, который позволяет различным программам взаимодействовать друг с другом. API определяет способы обмена данными и выполнения операций между различными компонентами программного обеспечения.

BGR (Blue, Green, Red) — это альтернативная модель представления цвета, которая является обратной к модели RGB. В модели BGR порядок следования основных цветов изменен на синий (Blue), зеленый (Green) и красный (Red). Модель BGR часто используется в некоторых библиотеках и инструментах компьютерного зрения, таких как OpenCV.

HTTP (Hypertext Transfer Protocol) — это протокол передачи данных в сети Интернет. HTTP используется для передачи информации между веб-серверами и клиентскими приложениями. Он определяет стандарты запросов и ответов, используемые для обмена данными в Интернете.

IoU (Intersection over Union) - это метрика, используемая в задачах компьютерного зрения для оценки качества детекции объектов. IoU вычисляется как отношение площади пересечения двух областей (bounding boxes) к их общей площади.

RGB (Red, Green, Blue) — это модель представления цвета, которая основана на комбинации трех основных цветов: красного (Red), зеленого (Green) и синего (Blue). Каждый цвет представлен числовым значением в диапазоне от 0 до 255, что позволяет создавать широкий спектр цветов путем смешивания основных цветов в различных пропорциях.

SADT (Structured Analysis and Design Technique) - это методика структурированного анализа и проектирования, которая используется в области информационных систем для анализа и моделирования бизнес-процессов. SADT предоставляет набор графических нотаций и инструментов для описания функций, данных и управления в рамках информационной системы.

SQL (Structured Query Language) — это язык программирования, используемый для работы с реляционными базами данных. SQL позволяет создавать, изменять, управлять и извлекать данные из баз данных. Он предоставляет различные операторы и команды для выполнения запросов к базам данных.

БД (База данных) — это организованная коллекция данных, которая обеспечивает эффективное хранение, управление и доступ к информации. Базы данных используются для хранения структурированных данных, обеспечивая централизованное хранение и обработку информации для различных приложений и пользователей.

ВВЕДЕНИЕ

Сегментация изображений на спутниковых снимках играет ключевую роль в области обработки и анализа геопространственных данных. Она представляет собой важный инструмент для автоматического распознавания объектов на земной поверхности и их классификации. Автоматизированный процесс сегментации спутниковых снимков позволяет эффективно анализировать большие территории и выявлять различные объекты, такие как здания, дороги, растительность и водные объекты.

Отказ от ручной сегментации в пользу автоматизированного подхода значительно ускоряет процесс обработки данных и снижает вероятность ошибок, связанных с человеческим фактором. Алгоритмы распознавания объектов на спутниковых снимках позволяют извлекать уникальные характеристики каждого объекта и использовать их для их точной локализации и классификации.

Распознавание объектов на изображениях включает в себя процесс нахождения и анализа конкретных объектов на изображении или видеопоследовательности. Путем извлечения характеристик объектов из обучающего набора данных и их сопоставления с объектами из тестового набора, сверточные нейронные сети могут точно определять и классифицировать объекты на изображениях.

Таким образом, обучение сверточных нейронных сетей для задачи распознавания объектов на спутниковых снимках имеет большое значение для эффективного анализа геопространственных данных и принятия важных решений в различных областях, таких как география, сельское хозяйство, градостроительство и экология.

1 ПОСТАНОВКА ЗАДАЧ РАЗРАБОТКИ

1.1 Описание предметной области

Сегментация объектов на спутниковых снимках представляет собой процесс разделения изображения на смысловые части или объекты. Этот процесс важен для различных прикладных задач, таких как мониторинг сельскохозяйственных угодий, управление лесными ресурсами, городское планирование, отслеживание изменений окружающей среды и др. Основная цель сегментации заключается в точном определении границ и классификации объектов, таких как здания, дороги, водоемы, растительность и другие элементы ландшафта.

Сегментация объектов на спутниковых снимках является активно развивающейся областью на пересечении компьютерного зрения и геоинформатики. Традиционные методы сегментации, такие как пороговая обработка и кластеризация, уступают место более современным методам, основанным на глубоких нейронных сетях. С появлением глубокого обучения точность и надежность сегментации значительно возросли. Современные модели, такие как Fully Convolutional Networks (FCN), U-Net и DeepLab, успешно применяются для сегментации медицинских изображений, а также спутниковых снимков.

1.2 Цели и задачи

Целью данной работы является разработка сервиса для автоматической сегментации объектов на спутниковых снимках. Сервис должен обеспечивать распознавание и классификацию различных типов объектов.

Задачи исследования:

* Анализ существующих методов сегментации: Изучение современных подходов и алгоритмов, применяемых для сегментации объектов на спутниковых снимках
* Сбор и подготовка данных: Создание обширной базы данных спутниковых снимков с аннотациями для обучения и тестирования модели.
* Разработка модели сегментации: Выбор и адаптация алгоритмов машинного обучения и компьютерного зрения для задачи сегментации
* Создание пользовательского интерфейса и программного интерфейса (API) для взаимодействия с сервисом.

1.3 Сравнение с аналогами

1.3.1 Формирование критериев оценки

Выбран ряд критериев:

1. Точность сегментации
2. Скорость обработки
3. Удобство использования
4. Возможность работы в команде
5. Надежность и стабильность

Воспользуемся методом базового критерия для вычисления коэффициентов важности локальных критериев [1]. Разделим критерии на 2 группы важности. В первую войдут точность сегментации, возможность работы в команде и надежность, во вторую (несущественную) – удобство использования и скорость обработки. Критерии первой группы превосходят критерии второй в k1 = 2 раза.

Рассчитаем нормировочный коэффициент по формуле:

Где ni – количество локальных критериев, которые входят в состав i-й группы,

g – количество групп важности локальных критериев,

βi – коэффициент, который показывает степень превосходства критериев

Получаем α1 = 0,25, α2 = 0,125 – весовые коэффициенты критериев для 1 и 2 группы соответственно.

В таблице 1 приведены критерии и их веса.

Таблица 1 – Коэффиценты важности критериев

|  |  |  |  |
| --- | --- | --- | --- |
| Код фактора | Название критерия | Группа важности | Вес критерия |
| X1 | Точность сегментации | 1 | 0,25 |
| Х2 | Скорость обработки | 2 | 0,125 |
| X3 | Удобство использования | 2 | 0,125 |
| Х4 | Возможность работы в команде | 1 | 0,25 |
| Х5 | Надежность и стабильность | 1 | 0,25 |

1.3.2 Варианты сервисов сегментации

Для сравнения были выбраны следующие сервисы сегментации:

1. Google Earth Engine
2. Esri ArcGIS
3. ENVI
4. Orfeo Toolbox
5. Shape Extract (разрабатываемый сервис)

В таблице 2 приведены варианты и значения коэффицентов.

Таблица 2 – Варианты сервисов сегментации

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Код фактора | Критерий | B1 | B2 | B3 | B4 | B5 |
| X1 | Точность сегментации | Высокая | Высокая | Хорошая | Высокая | Высокая |
| X2 | Скорость обработки | Высокая | Средняя | Средняя | Высокая | Средняя |
| X3 | Удобство использования | Высокое | Среднее | Высокое | Среднее | Высокое |
| X4 | Возможность работы в команде | Есть | Есть | Есть | Нет | Есть |
| X5 | Надежность и стаблильность | Средняя | Высокая | Средняя | Высокая | Высокая |

В таблице 3 приведены результаты попарного сравнения вариантов с целью отбора среди них Парето-оптимальных.

Таблица 3 – Сравнение вариантов на Парето-оптимальность

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | B1 | B2 | B3 | B4 | B5 |
| B1 | - | 0 | 1 | 0 | 0 |
| B2 | 0 | - | 0 | 0 | 0 |
| B3 | 0 | 0 | - | 0 | 0 |
| B4 | 0 | 1 | 0 | - | 0 |
| B5 | 0 | 1 | 1 | 0 | - |
| Парето-оптимальность варианта | Да | Нет | Нет | Да | Да |

В результате сравнения видим, что вариант B2 ни по одному критерию не превосходит варианты B4 и B5, а B3 не превосходит B1 и B5. Исключаем эти варианты из сравнения.

Для перевода качественных параметров в количественные применим вербально-числовую шкалу для критериев (см. таблицы 4, 5)

Таблица 4 – Вербально числовая шкала для критериев X1, X2, X3, X5

|  |  |
| --- | --- |
| Качественная оценка | Количественная оценка |
| Высокое | 1 |
| Среднее | 0,5 |
| Низкое | 0,1 |

Таблица 5 – Вербально числовая шкала для критерия X4

|  |  |
| --- | --- |
| Качественная оценка | Количественная оценка |
| Есть | 1 |
| Нет | 0,1 |

Сравним три варианта по пяти критериям (таблица 6).

Таблица 6 – Значение критериев для вариантов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Код критерия | Критерий | Значения | | | Вес критерия |
| B1 | B4 | B5 |
| X1 | Точность сегментации | 1 | 1 | 1 | 0,25 |
| X2 | Скорость обработки | 1 | 1 | 0,5 | 0,125 |
| X3 | Удобство использования | 1 | 0,5 | 1 | 0,125 |
| X4 | Возможность работы в команде | 1 | 0,1 | 1 | 0,25 |
| X5 | Надежность и стабильность | 0,5 | 1 | 1 | 0,25 |

1.3.3 Выбор лучшего варианта

Вычислим интегральный критерий методом максимума взвешенной суммы локальных критериев, используя формулу:

где n – число показателей сравнения;

m – количество вариантов сравнения;

– коэффициент важности *i*-го параметра сравнения;

– коэффициент нормализации.

Произведем нормализацию значений критериев и найдем значение интегрального критерия [2].

При нормализации критериев типа «чем больше, тем лучше» коэффициент нормализации рассчитываются по формуле:

где – значение *i*-го локального критерия, соответствующее максимальному значению среди сравниваемых вариантов решения.

Получим таблицу 7 с результатами расчётов

Таблица 7 – Расчет интегрального критерия (метод максимума взвешенной суммы локальных критериев)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код фактора | Весовой коэффициент | Значения показателей | | |
| В1 | В4 | В5 |
| Х1 | 0,25 | 1 | 1 | 1 |
| Х2 | 0,125 | 1 | 1 | 0,5 |
| Х3 | 0,125 | 1 | 0,5 | 1 |
| Х4 | 0,25 | 1 | 0,1 | 1 |
| Х5 | 0,25 | 0,5 | 1 | 1 |
|  | | 0,875 | 0,4625 | 0,9375 |

Получаем B5 (Сервис ShapeExtract)

Практическая ценность разработанного сервиса заключается в его способности автоматизировать и упростить анализ спутниковых снимков, что может существенно снизить затраты и повысить эффективность в таких областях, как сельское хозяйство, экология, урбанистика и др. Научная новизна работы заключается в разработке и применении новых методов и подходов для повышения точности и надежности сегментации объектов на спутниковых снимках, а также в интеграции лучших практик и технологий для создания высокоэффективного сервиса.

1.4 Типы задач сегментации

Задачи сегментации изображения можно разделить на типы на основе количества и типа информации, которую они передают. (см. рис.1)

* Семантическая сегментация.
* Сегментация экземпляра
* Паноптическая сегментация

Семантическая сегментация относится к классификации пикселей изображения по семантическим классам. Пиксели, принадлежащие к определенному классу, просто относятся к этому классу без учета другой информации или контекста [3]. Сегментация работает со множеством объектов одного класса как с единым целым.

Модели сегментации экземпляров классифицируют пиксели по категориям на основе «экземпляров», а не классов. Алгоритм сегментации экземпляров не знает к какому классу принадлежит классифицируемая область, но может разделить перекрывающиеся или очень похожие области объекта на основе их границ.

Паноптическая сегментация, самая последняя разработанная задача сегментации, может быть выражена как комбинация семантической сегментации и сегментации экземпляров, при которой каждый экземпляр объекта на изображении выделяется и предсказывается идентичность объекта.

На рисунке 1 наглядно видно различие между описанными тремя типами сегментации.

Изображение выглядит как текст, снимок экрана, на открытом воздухе, пляж

Автоматически созданное описание

Рисунок 1 – Типы сегментации

1.5 Семантическая сегментация

Как было сказано ранее семантическая сегментация заключается в разбиении фото или видеокадра на несколько сегментов, где каждому пикселю присваивается метка класса. Например, если на изображении присутствует автомобиль, сначала нужно отделить его от фона, определить границы объекта, а затем отметить каждый пиксель меткой «автомобиль». Далее, с помощью модели машинного обучения, пиксели одного класса группируются вместе.

В реальной жизни это работает следующим образом: у нас есть фотография улицы. Семантическая сегментация позволяет присвоить пикселям дороги метку «дорога», пикселям неба — метку «небо» и так далее. В результате каждый пиксель получает метку. Это помогает компьютеру лучше понимать содержимое изображения для принятия правильных решений на основе точных данных. Эта технология применяется в обучении беспилотного транспорта, чтобы он мог «видеть» и распознавать дорожные знаки, пешеходов и другие автомобили [4].

Также технология применяется при обработке аэрофотоснимков дроны часто задействуются для обследования территорий в условиях чрезвычайных ситуаций, таких как наводнения или землетрясения. Их задача — обнаружить живых людей и животных и передать эту информацию спасателям, используя технологии семантической сегментации.

Геозондирование включает использование спутниковых изображений для анализа и классификации земной поверхности. Это позволяет отслеживать процессы обезлесения, выявлять изменения в городской застройке и планировать инфраструктурные проекты, также с помощью семантической сегментации.

Технология находит применение в точном земледелии, где роботы, оснащенные камерами и сенсорами, определяют местоположение сельскохозяйственных культур и сорняков, чтобы проводить прополку в реальном времени. Зная точное расположение сорняков, роботы могут применять гербициды только на нужных участках, что снижает использование химических веществ и уменьшает негативное воздействие на окружающую среду.

1.6 Архитектуры нейронных сетей для решения задачи семантической сегментации

Существует множество архитектур нейронных сетей, специально разработанных для решения задачи семантической сегментации в области компьютерного зрения. Эти архитектуры используются для точного определения границ и классификации объектов на изображениях. Вот некоторые из наиболее популярных архитектур нейронных сетей для задачи сегментации:

1. U-Net является одной из самых широко используемых архитектур для семантической сегментации. Она обладает специфической структурой кодировщика-декодера, которая позволяет эффективно извлекать признаки и восстанавливать сегментированные изображения.
2. DeepLab представляет собой архитектуру, основанную на сверточных нейронных сетях с использованием аттенции и пирамидальных модулей для улучшения сегментации объектов на изображениях.
3. SegNet — еще одна популярная архитектура для семантической сегментации, которая активно использует методы агрегации признаков и декодирования для точного определения границ объектов.
4. DeepLabV3+ — это модель для сегментации изображений, которая сочетает в себе методы глубокого сверточного кодирования и декодирования, а также используя пространственные пирамидальные пулы и расширенные сверточные слои для точного распознавания объектов на изображениях.
5. PSPNet (Pyramid Scene Parsing Network) – использует пирамидальные модули для улучшения сегментации сцен на изображениях и повышения точности классификации объектов.
6. Архитектура LinkNet также является значимой в области семантической сегментации изображений. LinkNet была разработана с целью обеспечить высокую скорость обучения и высокую точность сегментации. Ее особенности включают в себя использование энкодера-декодера с промежуточными связями, что позволяет эффективно передавать информацию между слоями и избежать потери деталей при сегментации. Архитектура LinkNet также известна своей способностью эффективно работать с различными типами данных и условиями освещения, что делает ее привлекательным выбором для разнообразных задач сегментации изображений. Комбинация скорости обучения и высокой точности делает LinkNet популярным выбором для многих проектов в области компьютерного зрения и обработки изображений.

Каждая из этих архитектур имеет свои уникальные особенности и преимущества, что делает их эффективными инструментами для решения задачи семантической сегментации в области компьютерного зрения.Выбор конкретной архитектуры зависит от требований задачи, особенностей данных и доступных ресурсов для обучения и развертывания модели. Рассмотрим некоторые из них подробнее в следующих разделах.

1.6.1 U-Net

U-Net является одной из основных архитектур сверточных нейронных сетей (СНС) для задач сегментации изображений. Применяется не только для определения класса изображения, но и для разделения на области и создания маски для каждого класса. U-Net – очень быстрая сеть, способная сегментировать изображение 512x512 менее чем за секунду.

Архитектура состоит из стягивающего пути для захвата контекста и симметричного расширяющегося пути, который позволяет осуществить точную локализацию (см. рис. 2). [5] То есть стягивающий путь содержит слои, фиксирующие информацию с изображений, уменьшающие пространственное разрешение входных данных, а расширяющийся путь содержит слои, декодирующие закодированные данные, определяющие расположение объектов и создающие маски сегментации, используя информацию, полученную на стягивающем пути.

Рисунок 2 демонстрирует преобразование изображение размером 572 на 572 в оттенках градации серого в двоичную сегментированную выходную карту размером 388×388×2. Размер выходных данных меньше размера входных данных, потому что не используется заполнение. Однако, если использовать “padding”, можно сохранить размер входных данных.

В процессе сжатия пути входное изображение постепенно уменьшается по высоте и ширине, но увеличивается количество каналов. Это увеличение каналов позволяет сети захватывать объекты высокого уровня по мере продвижения по пути. В узком месте выполняется заключительная операция свертки для создания карты объектов размером 30 × 30 × 1024. Затем расширенный путь извлекает карту объектов из узкого места и преобразует ее обратно в изображение того же размера, что и исходные входные данные. Это делается с использованием слоев с повышающей дискретизацией, которые увеличивают пространственное разрешение карты объектов при одновременном сокращении количества каналов. Наконец, каждый пиксель на выходном изображении представляет собой метку, соответствующую определенному объекту или классу на входном изображении. [6]

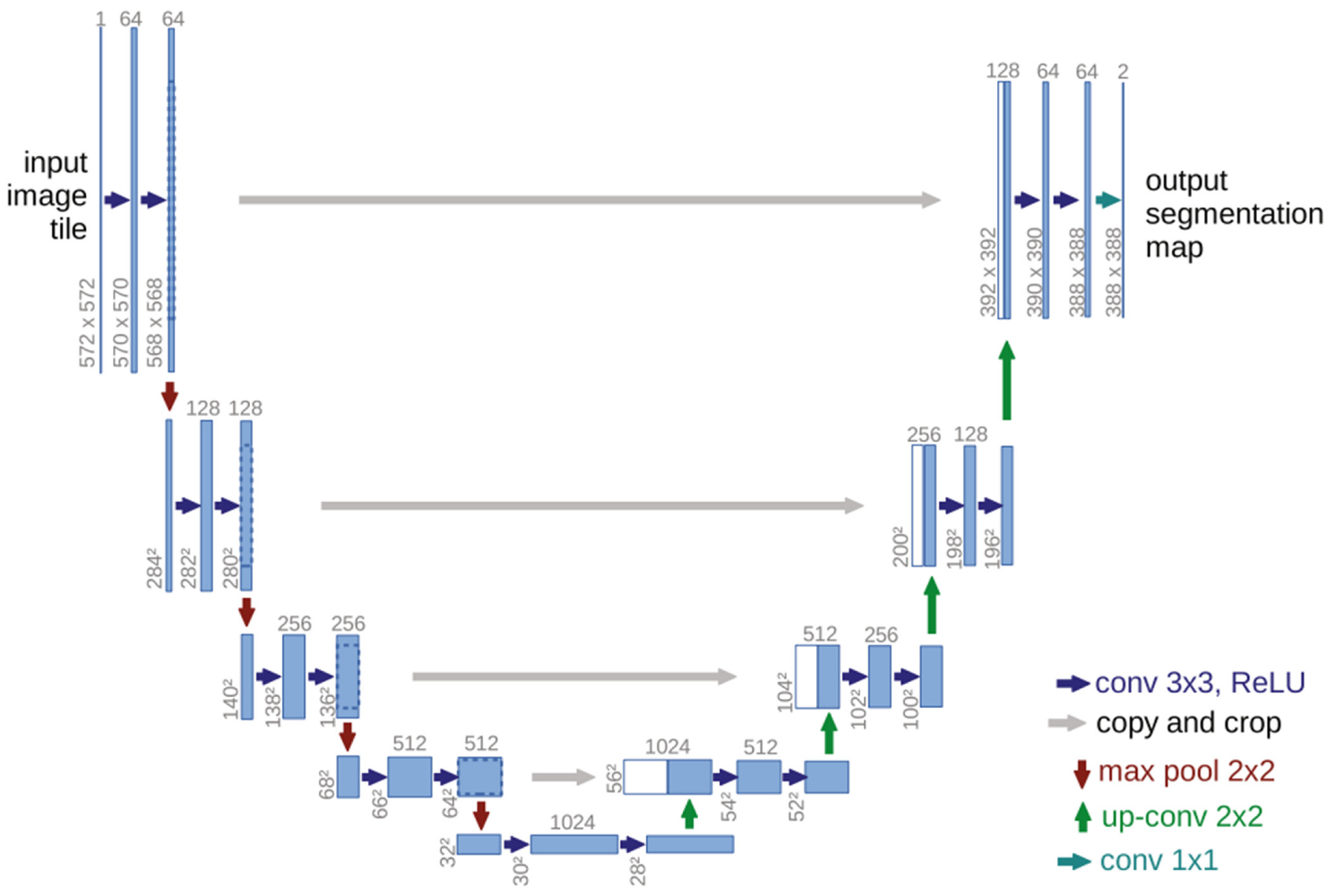


Рисунок 2 – Архитектура U-Net

1.6.2 DeepLabV3Plus

DeepLabV3 — это глубокая нейронная сеть, разработанная для задач семантической сегментации, которая позволяет классифицировать каждый пиксель изображения и разделять его на различные классы. Основные концепции, заложенные в эту архитектуру, включают использование Atrous (Dilated) Convolutions и Atrous Spatial Pyramid Pooling (ASPP), что обеспечивает эффективное извлечение мультимасштабных признаков и высокую точность сегментации.

Глубокие и полностью сверточные нейронные сети (Fully Convolutional Networks, FCNs) зарекомендовали себя как эффективные инструменты для задач сегментации изображений. Эти сети способны захватывать детализированные карты признаков, которые содержат богатую информацию на дальние расстояния. Это позволяет достигать высокой точности при разделении изображения на сегменты.

Ключевым компонентом архитектуры DeepLabV3 является Atrous Convolution (или Dilated Convolution). Этот метод позволяет извлекать более плотные признаки на глубоких уровнях сети, не увеличивая количество параметров. Atrous Convolution использует "дыры" или пропуски между элементами фильтра, что позволяет контролировать размер рецептивного поля и разрешение карты признаков, сохраняя при этом вычислительную эффективность.

Архитектура DeepLabV3 включает в себя два основных компонента: основной блок (backbone) и голову (head). Backbone отвечает за кодирование изображения и извлечение признаков, в то время как head выполняет декодирование и классификацию пикселей. Один из методов, применяемых в этой архитектуре, называется Multi-Grid. Он позволяет использовать различные скорости атрофной свертки в разных блоках сети, что дополнительно повышает способность сети захватывать признаки на разных масштабах. [7]

1.6.3 Link-Net

Архитектура LinkNET содержит серию блоков кодера и декодера, которые, соответственно, разбивают изображение и восстанавливают его. В конце изображения проходят через несколько сверточных слоев. Дизайн модели направлен на минимизацию количества параметров, что делает модель подходящей для задач сегментации в реальном времени.

Архитектура LinkNet была разработана на основе архитектуры ResNet. Архитектура LinkNet34, используемая для сегментации ядер, представлена на рисунке 3.

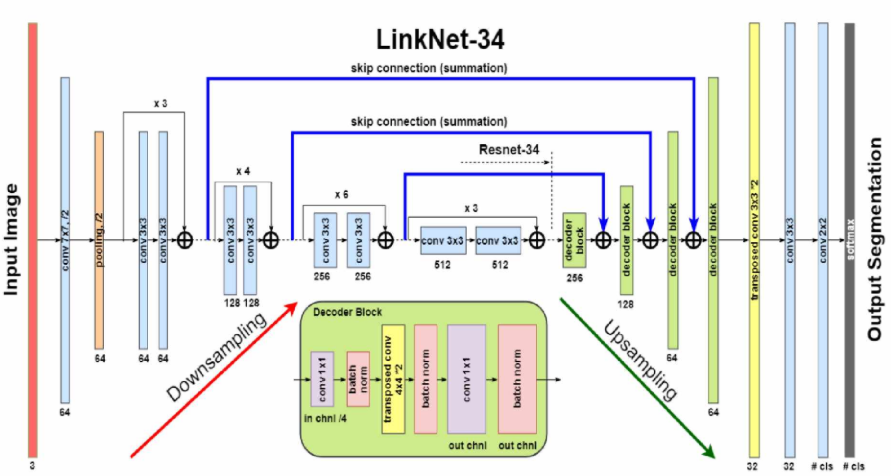


Рисунок 3 – Схематическое изображение архитектуры LinkNET-34

Первый блок модели выполняет свертку на входном изображении с использованием ядра размером 7x7 с шагом 2. Затем следует слой макспулинга с шагом 2. Следующий набор слоев состоит из серии остаточных блоков. В каждом остаточном блоке начальный сверточный слой будет иметь шаг 2, выполняя подвыборку входных данных, а остальные сверточные слои в соответствующем блоке будут иметь шаг 1. Также модель состоит из серии блоков декодера, которые связаны с соответствующим блоком кодера. Блок декодера выполняет свертку 1x1, за которой следует нормализация по батчу и транспонированные свертки для повторного масштабирования полученных карт признаков в предыдущих блоках.

Архитектура LinkNET значительно отличается от существующих нейронных архитектур, разработанных для сегментации на уровне пикселей. Уникальность архитектуры LinkNET заключается в связи между блоком кодера и соответствующим блоком декодера. Обычно пространственная информация теряется во время последовательности операций декодирования, и её сложно восстановить из выходных данных кодера. Посредством индексов пула кодер связывается с декодером, и эти параметры являются нетренируемыми. Некоторые другие методы напрямую связывают выходные данные кодера с декодером для выполнения сегментации. В архитектуре LinkNET создается связь, чтобы обойти вход кодера и подать его на выход декодера. Эта связь направлена на восстановление утраченной пространственной информации из-за последовательности операций кодирования и используется при повторном масштабировании декодером. Согласно этой архитектуре, декодер использует меньше параметров, так как он делится знаниями, полученными кодером. Эта архитектура помогает создавать более эффективную сеть, подходящую для операций в реальном времени по сравнению с существующими архитектурами, используемыми для сегментации [8].

1.6.4 Сравнение архитектур

В таблице 8 представлены сравнительные характеристики методов для сегментации изображений, описанных выше.

Таблица 8 – Сравнение методов для сегментации изображений

| **№** | **Архитектура** | **IoU** | **PixelAccuracy** | **Время обучения, мин** |
| --- | --- | --- | --- | --- |
| 1 | U-Net­ | 0.7150851425805498 | 0.7989174171730323 | 151.92 |
| 2 | LinkNet | 0.5739544702571099 | 0.7007950676812065 | 76.95 |
| 3 | DeepLabV3Plus | 0.75717711697332 | 0.8103158738878038 | 129.79 |

Сравнение трех архитектур для сегментации изображений — U-Net, LinkNet и DeepLabV3Plus — показывает, что каждая из них имеет свои сильные и слабые стороны.

Во-первых, DeepLabV3Plus демонстрирует наилучшие показатели среди всех трех архитектур. Со значением IoU (Intersection over Union) 0.757 и PixelAccuracy (точность по пикселям) 0.810, эта архитектура превосходит U-Net и LinkNet. Кроме того, время обучения для DeepLabV3Plus составляет 129.79 минут, что делает ее второй по скорости после LinkNet.

2 ОБУЧЕНИЕ МОДЕЛИ DEEPLABV3PLUS

Ранее мы провели сравнение трех архитектур моделей для сегментации объектов на изображениях. С учётом анализа мы используем архитектуру DeepLabV3Plus для реализации задачи сегментации спутниковых снимков. Данная архитектура показала потенциал в эффективности обучения модели и может быть ключевым инструментом для создания нашего веб-сервиса по обработке изображений.

2.1 Набор данных

Для обучения модели воспользуемся размеченным набором данных. Набор данных состоит из аэрофотоснимков Дубая, полученных со спутников MBRSC и аннотированных с помощью попиксельной семантической сегментации по 6 классам. Общий объем набора данных — 72 изображения.

Классы и соответствующие им цвета на масках приведены в таблице 9.

Таблица 9 – Соответствие классов и цветов на масках

| **Класс** | **Цветовой индекс** | **Цвет** |
| --- | --- | --- |
| Здание | #3C1098 |  |
| Земельный участок | #8429F6 |  |
| Дорога | #6EC1E4 |  |
| Растительность | #FEDD3A |  |
| Вода | #E2A929 |  |
| Без маркировки | #9B9B9B |  |

На рисунке 4 представлен пример изображения, снятого со спутника, и соответствующей ему маске. Изображение содержит озеро, деревья, дороги, здания, землю, а маска содержит соответствующие классы, приведенные в таблице 2.

A collage of maps of land and water

Description automatically generated

Рисунок 4 – Пример изображения и соответствующей ему маски

2.2 Предобработка и преобразования

До передачи данных в модель нейронной сети для обучения нужно их подготовить. Для начала переведем HEX-коды цветов в значения RGB и назначим каждому классу метки. Получим значения, указанные в таблице 10.

Таблица 10 – Метки классов

| **Метка класса** | **Класс** | **HEX** | **RGB** | **Цвет** |
| --- | --- | --- | --- | --- |
| 0 | Здание | #3C1098 | (226, 169, 41) |  |
| 1 | Земельный участок | #8429F6 | (132, 41, 246) |  |
| 2 | Дорога | #6EC1E4 | (110, 193, 228) |  |
| 3 | Растительность | #FEDD3A | (60, 16, 152) |  |
| 4 | Вода | #E2A929 | (254, 221, 58) |  |
| 5 | Без маркировки | #9B9B9B | (155, 155, 155) |  |

Реализуем класс Aeroscapes\_Dataset, который будет представлять наш датасет. Тогда функция \_\_getitem\_\_(self, index) будет считывать изображение и маску и конвертировать их из цветового пространства BGR в RGB, т.к. используется библиотека OpenCV, в коорой цветовое пространство по умолчанию - BGR (синий, зеленый, красный). Затем маска преобразуется из формата RGB в маску, использующую индексы классов, согласно таблице 3. После этого изображение преобразуется в тензоры, нормализуется с использованием средних и стандартных отклонений, а маска из массива NumPy преобразуется в тензор PyTorch типа long. Таким образом функция вернет 2 тензора: тензор маски и тензор изображения.

Определим наборы преобразований t\_train и t\_val для обучающего и валидационного наборов данных. Набор преобразований включает в себя изменение размера изображений, отражения по горизонтали и вертикали, искажения сетки, случайное изменение яркости и контраста, а также добавление гауссовского шума.

Затем создадим объекты ранее упомянутого класса Aeroscapes\_Dataset для обучающего и валидационного наборов данных, указывая пути к изображениям, маскам, индексы обучающих данных, средние и стандартные значения для нормализации, набор преобразований.

Наконец, создадим загрузчики данных DataLoader для обучающего и валидационного наборов данных с указанным размером пакета и перемешиванием данных. Эти загрузчики данных используются для эффективной передачи данных в процессе обучения модели.

2.3 Модель

2.3.1 Создание модели

Создадим модель сегментации изображений с использованием архитектуры DeepLabV3Plus, который использует предварительно обученную MobileNetV2 в качестве энкодера.

Передаем следующие параметры:

* 'mobilenet\_v2' – указывает на использование архитектуры MobileNetV2 в качестве энкодера модели,
* encoder\_weights='imagenet' – загружает предобученные веса энкодера из набора данных ImageNet для инициализации модели,
* classes=6 – устанавливает количество классов для задачи сегментации на 6,
* activation=None – отключает функцию активации на выходном слое модели,
* encoder\_depth=5 – указывает на глубину энкодера, то есть количество слоев энкодера,
* decoder\_channels=[256, 128, 64, 32, 16] – Создаем модель Unet с использованием архитектуры MobilenetV2 (MobileNetV2 — это легкая сверточная нейронная сеть, которая хорошо подходит для решения задач компьютерного зрения при ограниченных ресурсах.) в качестве кодера.

2.3.2 Метрики

Для оценки точности модели были реализованы следующие метрики.

Intersection over Union (IoU) — это мера, используемая для оценки качества сегментации объектов на изображениях. Она вычисляется как отношение пересечения между предсказанным и истинным масками к их объединению. Она позволяет оценить, насколько хорошо модель выделяет объекты на изображении и правильно определяет границы объектов.

Реализуем функцию для вычисления среднего значения Intersection over Union (IoU). Исходное изображение и его сегментированная версия передаются в функцию в виде двух масок: pred\_mask - предсказанная моделью маска, и mask - истинная маска. В первую очередь, предсказанная маска приводится к вероятностному распределению классов с помощью функции softmax. Затем выбирается индекс класса с наибольшей вероятностью для каждого пикселя, чтобы получить бинарную маску предсказанных классов.

Далее, для каждого класса выполняется следующее:

1. Создаются бинарные маски для предсказанных и истинных классов.
2. Проверяется, содержит ли истинная маска пиксели текущего класса. Если нет, то значение IoU для этого класса устанавливается как NaN.
3. Иначе, вычисляется количество пикселей, которые принадлежат и одновременно предсказанному, и истинному классу, а также общее количество пикселей, принадлежащих любому из этих классов.
4. IoU для текущего класса вычисляется по формуле, включающей количество пересечений и объединений пикселей с учетом плавающего параметра smooth для избежания деления на ноль.
5. Полученные значения IoU для каждого класса добавляются в список.

Функция возвращает среднее значение IoU по всем классам, игнорируя NaN значения.

Пиксельная точность (pixel accuracy) представляет собой простую метрику, показывающую долю правильно классифицированных пикселей в изображении. Она позволяет оценить общую точность модели в сегментации объектов по пикселям.

Реализуем функцию для оценки пиксельной точности.

Сначала, вывод модели, представляющий вероятности принадлежности к различным классам, преобразуется в конкретные классы путем выбора класса с наибольшей вероятностью для каждого пикселя. Это позволяет нам получить предсказанные классы для каждого пикселя на изображении.

Затем, эти предсказанные классы сравниваются с истинными классами, представленными в маске. После чего для каждого пикселя определяется, правильно ли модель классифицировала его. Результатом этого сравнения является бинарная маска, в которой значение 1 обозначает правильно классифицированный пиксель, а значение 0 - неправильно классифицированный.

Наконец, точность вычисляется как отношение числа правильно классифицированных пикселей ко всем пикселям на изображении. Это достигается путем подсчета суммы значений бинарной маски и деления этой суммы на общее количество пикселей на изображении.

Таким образом, результатом выполнения этой функции будет доля правильно классифицированных пикселей на изображении, что представляет собой простую и понятную метрику качества сегментации изображений.

Описанные метрики являются стандартными метриками для задачи сегментации. Использование этих метрик помогает отслеживать прогресс обучения модели и принимать решения о ее улучшении на основе конкретных численных показателей.

2.3.3 Обучение модели

Будем обучать модель на данных из train\_loader и оценивает на данных из val\_loader. Установим максимальную скорость обучения = 10-4, будем обучать модель в течение 15 эпох. Зададим условие, что обучние останавливается, когда потери не уменьшаются в течение 6 эпох.

В результате обучение остановилось на 14 эпохах и мы получили следующие графики, изображенные на рисунках 5, 6, 7.

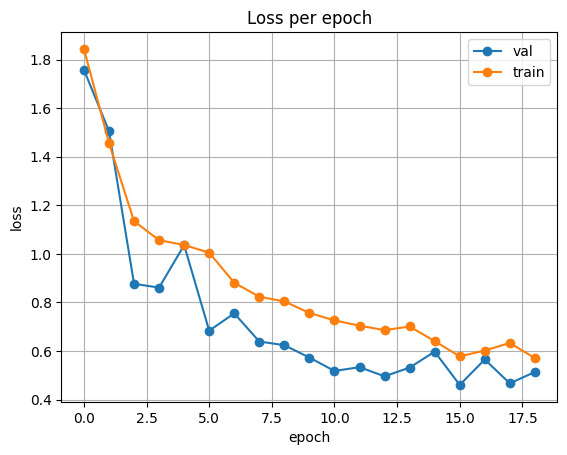


Рисунок 5 – График потерь

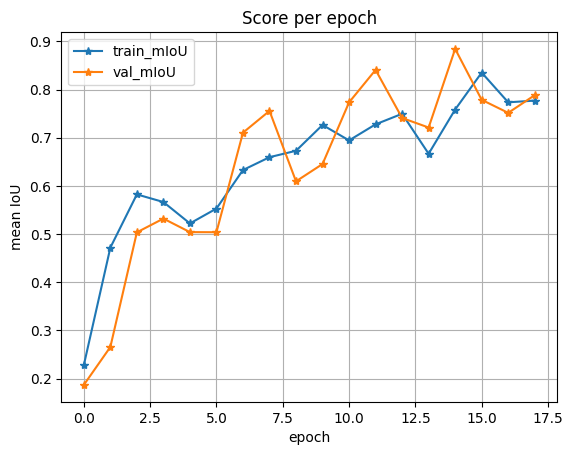


Рисунок 6 – График среднего значения метрики IoU

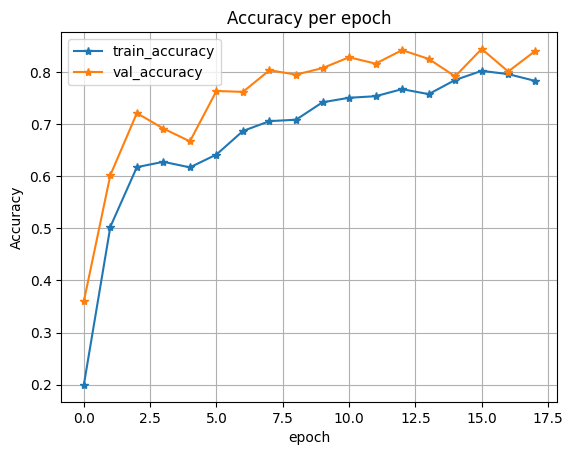


Рисунок 7 – График точности

В результате работы модели получаем сегментированное изображение (см. рис. 8).

 Изображение выглядит как снимок экрана, карта, искусство

Автоматически созданное описание

Рисунок 8 – Результат сегментации

Сохраняем модель для дальнейшего использования в веб-сервисе.

3 СЕРВИС СЕГМЕНТАЦИИ

Сервис сегментации ShapeExtract – сервис сегментации, который использует модель машинного обучения для выделения форм объектов на изображениях. Для сегментации объектов на изображениях применяется архитектура U-Net.

Процесс работы сервиса ShapeExtract включает в себя загрузку изображения, которое подается на вход модели. Таким образом, пользователь может легко определить форму и контуры объектов на изображении с помощью данного сервиса.

3.1 Постановка задачи

Сервис сегментации объектов на спутниковых снимках рассматривается с позиции пользователя. Применение этой системы позволяет эффективно анализировать и использовать геопространственные данные.

Система сегментации объектов на спутниковых снимках включает несколько ключевых этапов. Сначала загружаются и получают спутниковые изображения, предназначенные для анализа. Далее система использует различные алгоритмы и методы сегментации, чтобы идентифицировать и выделить на снимках разные объекты, такие как здания, дороги, водные поверхности и растительность. Затем результаты сегментации визуализируются: сегментированные объекты отображаются непосредственно на изображении или создается отдельная визуализация с отмеченными объектами. На последнем этапе обработанные изображения сохраняются и становятся доступными для дальнейшего использования, анализа или передачи другим пользователям и системам.

Это позволяет проводить эффективный анализ геопространственных данных, выявлять изменения в окружающей среде, отслеживать развитие инфраструктуры и применять эти данные для прогнозирования и планирования в различных областях, таких как геоинформационные системы, геопланирование и экология.

Функции системы представлены в таблице 11.

Цель: описать работу сервиса сегментации объектов на спутниковых снимках.

Точка зрения: пользователь.

Таблица 11 – Описание основных функций

| **Функции** | **Описание** | **Данные, требуемые для выполнения функций** |
| --- | --- | --- |
| 1. Загрузить изображение | Пользователь может загрузить спутниковое изображение в систему для дальнейшего анализа и сегментации | Входные данные:  Пользовательское изображение  Управление:  Правила работы с системой  Законодательство РФ  Механизмы:  Пользователь  Сервис |
| 1. Получить обработанное изображение | После обработки загруженного изображения пользователь получает доступ к сегментированному снимку с выделенными объектами, такими как здания, дороги, водные объекты и растительность | Входные данные:  Загруженное изображение  Управление:  Правила использования модели МО  Законодательство РФ  Механизмы:  Обученная модель МО  Пользователь  Сервис |
| 1. Сохранить изображение на устройстве | После обработки загруженного изображения пользователь может сохранить изображение на устройстве | Входные данные:  Обработанное изображение  Запрос на сохранение  Управление:  Правила использования модели МО  Законодательство РФ  Механизмы:  Обученная модель МО  Пользователь  Сервис |
| 1. Добавить изображение на страницу команды | Пользователь может добавить обработанное изображение на общую страницу команды для совместного использования и анализа | Входные данные:  Обработанное изображение  Управление:  Законодательство РФ  Механизмы:  Пользователь  Сервис |
| 1. Удалить изображение со страницы команды | Пользователь имеет возможность удалить ненужное или устаревшее изображение со страницы команды | Входные данные:  Изображение добавленное в команду  Управление:  Законодательство РФ  Механизмы:  Пользователь  Сервис |
| 1. Оставить комментарий под изображением | Пользователь может оставить комментарий под любым изображением, чтобы поделиться наблюдениями, задать вопросы или предложить изменения | Входные данные:  Изображение добавленное в команду  Управление:  Законодательство РФ  Механизмы:  Пользователь  Сервис |

3.2 Функциональная модель

Для изучения предметной области использовалась методология SADT (Structured Analysis and Design Technique). Функциональная модель SADT отображает функциональную структуру системы, включая выполняемые ею действия и связи между этими действиями. Процесс создания модели начинается с контекстной диаграммы, которая представляет всю систему в виде одной простой компоненты – блока «Сегментация объектов на спутниковых снимках», а также дуг, отображающих основные связи системы с внешней средой. Затем создается диаграмма декомпозиции, где контекстная диаграмма разбивается на отдельные активности, что позволяет выявить полный набор подфункций. Каждая подфункция представлена отдельным блоком с границами, определенными интерфейсными дугами.

Начинаем построение модели с контекстной диаграммы – блок «Сегментация объектов на спутниковых снимках» и стрелок, демонстрирующих взаимодействие между системой и внешним миром (см. рис. 9)

Второй уровень модели диаграмма декомпозиции, которая представляет из себя разбиение контекстно диаграммы и служит для выявления всех подфункций, где подфункция изображена в виде одного блока, а его границы определяются интерфейсными дугами. В данной системе в результате разбиения контекстной диаграммы были получены 5 блоков (см. рис. 10): «Загрузить изображение», «Получить обработанное изображение», «Сохранить изображение на устройстве», «Добавить изображение на страницу команды», «Удалить изображение со страницы команды», «Оставить комментарий пол изображением».

Пользователи могут начать с загрузки изображений на платформу для последующей обработки или просмотра. После обработки изображения система предоставляет возможность получить результат обработки. Затем пользователи могут добавить обработанные изображения на страницу своей команды для обсуждения и совместного использования. В случае необходимости, изображения можно удалить со страницы команды. Кроме того, система позволяет пользователям оставлять комментарии под изображениями, что способствует обсуждению и взаимодействию в рамках команды. Таким образом, данная система обеспечивает полный цикл работы с изображениями в командной среде, начиная с загрузки и обработки и заканчивая обсуждением и совместным использованием.

Третий уровень модели – диаграмма декомпозиции блока «Получить обработанное изображение».

Диаграмма декомпозиции блока «Получить обработанное изображение» (см. рис. 11) включает в себя следующие блоки:

1. Получить изображение с измененным размером - масштабирование, с целью получить изображение с требуемыми размерами
2. Получить предобработанное изображение - подготовка изображения к следующим этапам обработки
3. Получить сегментированное изображение – производится егментация изображения на классы
4. Получить постобработанное изображение - постобработка изображения включает в себя дополнительные операции обработки, которые могут быть применены после сегментации

Изображение выглядит как текст, диаграмма, Параллельный, число

Автоматически созданное описание

Рисунок 9 – Контекстная диаграмма сервиса сегментации объектов на спутниковых снимках

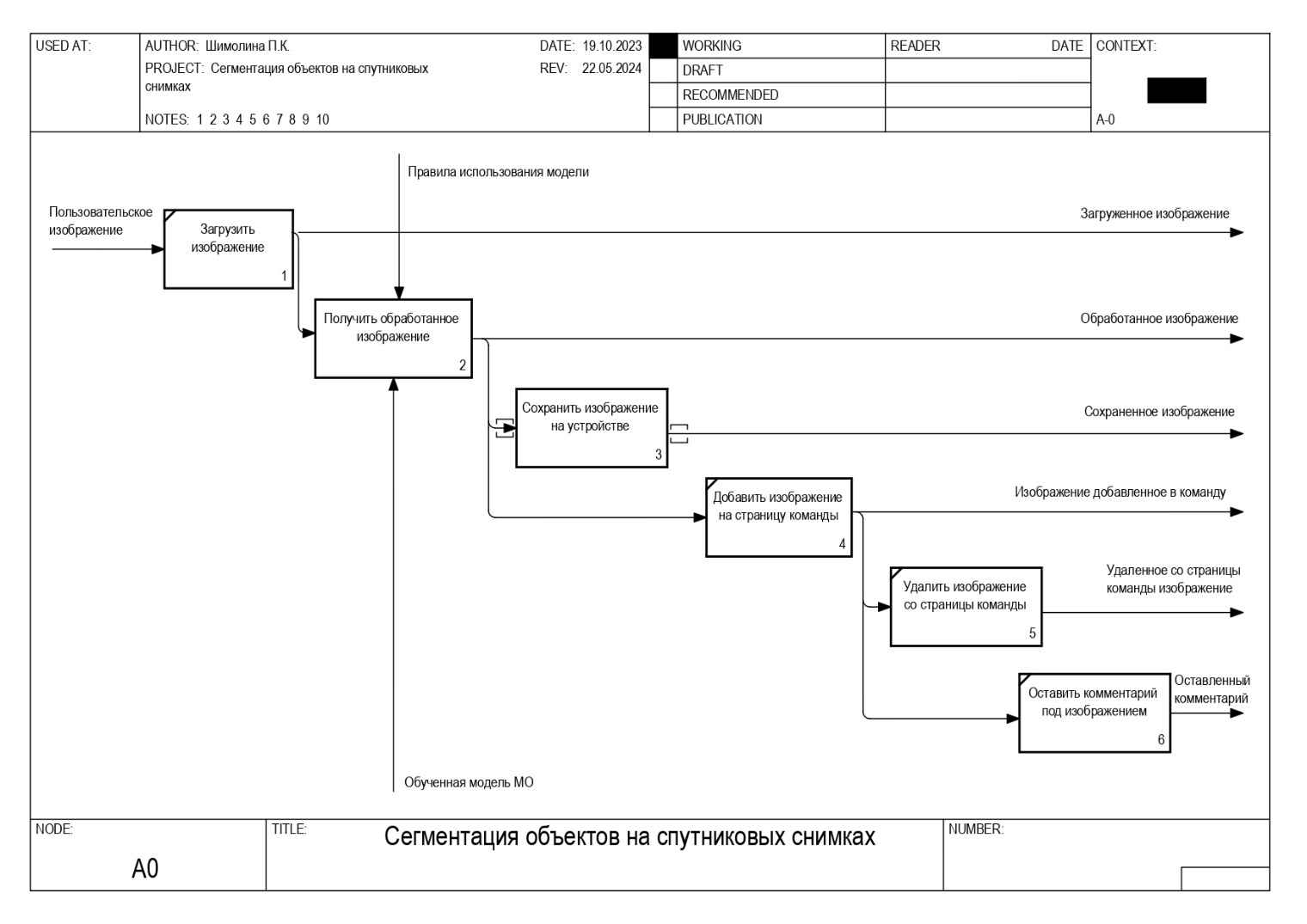


Рисунок 10 – Диаграмма декомпозиции сервиса сегментации объектов на спутниковых снимках

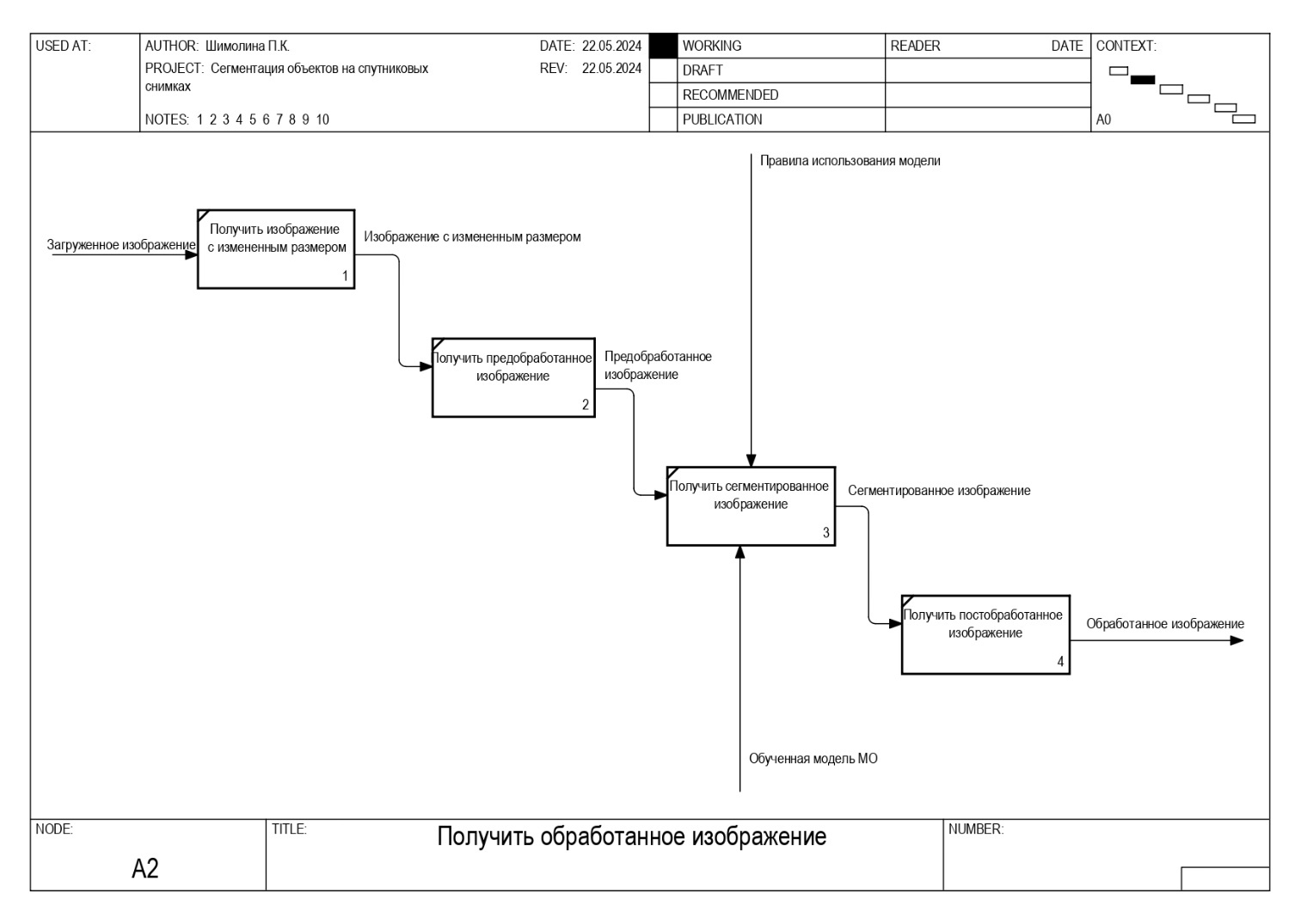


Рисунок 11 – Диаграмма функции «Получить обработанное изображение»

3.3 Диаграмма сущность-связь

Диаграмма сущность-связь отвечает за сущности системы и их взаимосвязь, содержит такие данные как идентификация сущностей, атрибутов и их связей.

Сущности диаграммы изображены в виде прямоугольника, содержащем её имя. Атрибуты сущности записаны внутри прямоугольника. Также определены ключевые атрибуты сущностей.Связи изображены линией, которая связывает две сущности, участвующие в отношении.

Такая диаграмма является методом представления информационной структуры базы данных в графическом виде для более простого и наглядного отображения основных компонентов конкретного проекта базы данных.

В диаграмме (см. рис. 12) представлено 7 сущностей: «Пользователь», «Загруженное\_изображение», «Сегментированное\_изображение», «Команда», «Пользователь\_Изображение», «Команда\_Изображение\_пользователя», «Комментарий».

Сущность «Пользователь» представляет собой пользователя и содержит информацию о имени пользователя, имени, фамилии, пароле, электронной почте и дате регистрации.

Сущность «Загруженное\_изображение» представляет изображение, загруженное пользователем, и содержит поле с изображением.

Сущность «Сегментированное\_изображение» представляет сегментированное изображение, полученное из загруженного, и содержит поле с изображением.

Сущность «Команда» содержит информацию об имени команды, ее описании, дате создания и идентификаторе создателя команды.

Сущность «Пользователь\_Изображение» содержит идентификаторы загруженного изображения, сегментированного изображения, пользователя, загрузившего фото и поле, информирующее о том, доступно ли изображение команде.

Сущность «Команда\_Изображение\_пользователя» содержит идентификатор изображения пользователя, идентификатор команды и дату, когда изображение было добавлено в команду.

Сущность «Комментарий» содержит идентификатор команды, время публикации, автора и текст комментария.

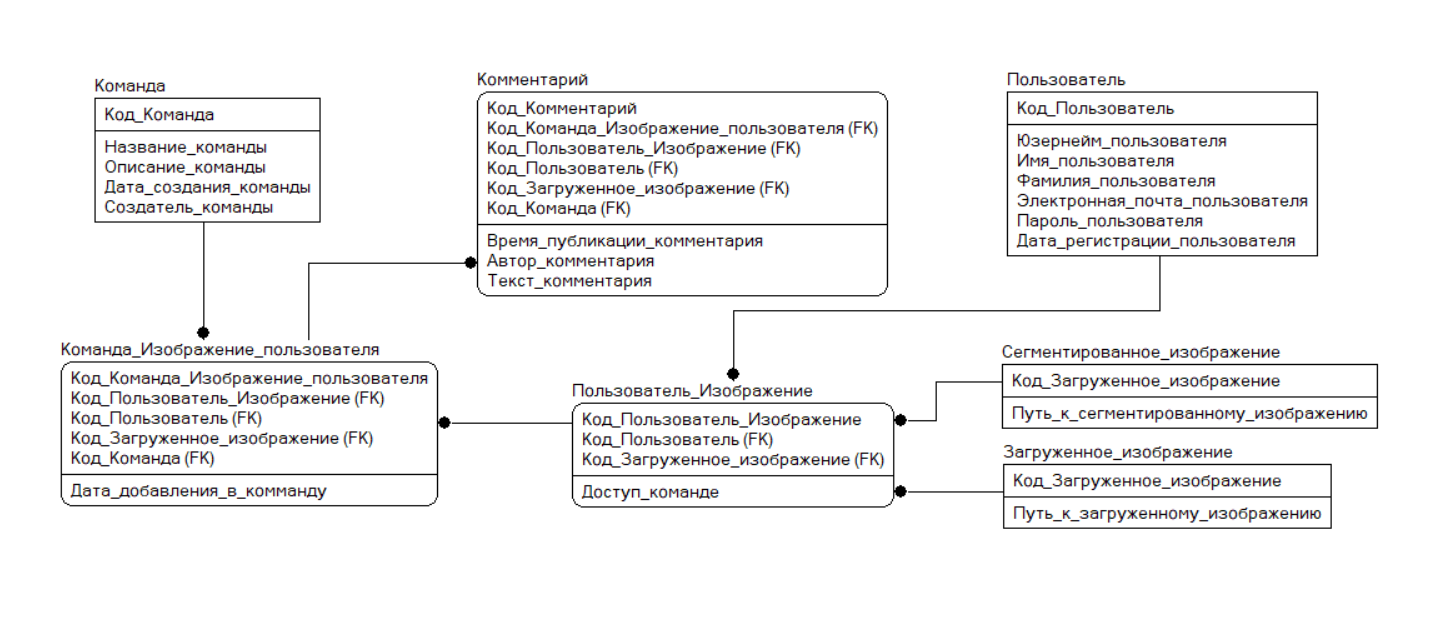


Рисунок 12 – Логическая модель данных

3.4 Используемые технологии

Для бэкенд-сервера был выбран фреймворк Django, который предоставляет удобные инструменты для создания веб-приложений на Python. Django обладает множеством возможностей, включая встроенный механизм аутентификации пользователей, защиту от типичных уязвимостей веб-приложений и многое другое. [9] Благодаря широкому сообществу разработчиков разработка веб-сервиса на Django эффективна и удобна.

Для хранения данных была выбрана база данных SQLite. SQLite — это внутрипроцессная библиотека, реализующая автономный, бессерверный, не требующий настройки транзакционный механизм базы данных SQL. [10] SQLite является легковесной и простой в использовании базой данных, которая не требует отдельного сервера. Она хорошо подходит для небольших и средних проектов, таких как веб-сервисы, благодаря своей простоте в настройке и использовании. SQLite также поддерживает стандарт SQL и обладает множеством функций, что делает её удобным выбором для многих приложений.

Для фронтенд-разработки был выбран JavaScript, один из самых популярных языков программирования для создания интерактивных веб-страниц. JavaScript позволяет создавать динамические элементы на веб-страницах, общаться с сервером без перезагрузки страницы (AJAX), а также обеспечивать взаимодействие с пользователем. Использование JavaScript позволит сделать веб-сервис более динамичным и привлекательным для пользователей.

Выбор этих технологий обоснован их совместимостью друг с другом, а также их способностью обеспечить удобную разработку, масштабируемость и безопасность веб-сервиса. Кроме того, использование Django позволяет легко интегрировать SQLite и JavaScript в проект, обеспечивая эффективную работу всей системы.

3.5 Разработка сервиса

При разработке веб-сервиса обычно выделяют несколько ключевых компонентов:

1. Бэкенд (Backend): это серверная часть веб-сервиса, которая отвечает за обработку запросов пользователя, взаимодействие с базой данных, бизнес-логику приложения.

2. Фронтенд (Frontend) – это та часть веб-сервиса, с которой взаимодействует пользователь. Фронтенд отвечает за отображение контента, интерактивные элементы, пользовательский интерфейс.

3. База данных (Database): это хранилище данных, используемое веб-сервисом для сохранения информации. Выбор базы данных зависит от требований проекта.

3.5.1 Backend

Бэкенд (Backend) — это серверная часть веб-приложения, которая отвечает за обработку запросов пользователя, взаимодействие с базой данных, бизнес-логику приложения и генерацию ответов, которые отображаются на фронтенде. Основными компонентами бэкенда обычно являются:

1. Сервер: это программное обеспечение, которое обеспечивает работу веб-сервиса, принимает запросы от клиентов и возвращает ответы. Сервер может быть написан на различных языках программирования и работать с использованием различных протоколов, таких как HTTP.

2. База данных: это хранилище данных, которое используется для сохранения информации, необходимой для работы приложения. База данных хранит данные в структурированном формате и обеспечивает доступ к ним через запросы.

3. Бизнес-логика: это часть приложения, которая определяет, как приложение должно обрабатывать запросы пользователя и выполнять определенные задачи. Бизнес-логика включает в себя правила и процессы, которые определяют логику работы приложения.

4. API (Application Programming Interface): это интерфейс, который определяет способы взаимодействия между различными компонентами приложения. API позволяет передавать данные и команды между клиентской и серверной частями приложения.

Взаимодействие между компонентами бэкенда обычно происходит следующим образом:

* Клиент отправляет запрос на сервер через сеть (например, через протокол HTTP).
* Сервер принимает запрос, обрабатывает его с помощью бизнес-логики и доступа к базе данных.
* Сервер формирует ответ на запрос и отправляет его обратно клиенту.
* Клиент получает ответ от сервера и отображает его пользователю на фронтенде.

3.5.1.1 Модели

Модели в Django представляют собой классы Python, которые отображают структуру базы данных. Они определяют, как данные будут храниться и взаимодействовать с базой данных. Модели позволяют создавать, обновлять, удалять и извлекать данные из базы данных, используя ORM (Object-Relational Mapping). Каждая модель соответствует таблице в базе данных и определяет поля, связи и методы для работы с данными.

При разработке сервиса были созданы следующие модели:

1. UploadedPhoto:

* photo: Поле типа ImageField, где хранится загруженная пользователем фотография

1. SegmentedPhoto

* photo: Поле типа ImageField, где хранится сегментированная версия фотографии

1. UserPhoto

* uploaded\_photo: Поле типа OneToOneField, связывает с моделью UploadedPhoto
* segmented\_photo: Поле типа OneToOneField, связывает с моделью SegmentedPhoto
* user: Поле типа ForeignKey, связывает с моделью User
* is\_visible\_to\_team: Логическое поле, указывающее, видна ли фотография команде

1. TeamPhoto

* user\_photo: Поле типа OneToOneField, связывает с моделью UserPhoto
* owner: Поле типа ForeignKey, связывает с моделью User и обозначает владельца фотографии
* team: Поле типа ForeignKey, связывает с моделью Team

1. Comment

* team\_photo: Поле типа ForeignKey, связывает с моделью TeamPhoto
* author: Поле типа ForeignKey, связывает с моделью User
* publication\_datetime: Поле типа DateTimeField, автоматически устанавливается на текущую дату и время при создании комментария
* text: Текстовое поле для хранения текста комментария

1. Team

* teamname: Поле типа CharField, хранит имя команды (максимальная длина 100 символов)
* description: Текстовое поле для описания команды
* created\_at: Поле типа DateTimeField, автоматически устанавливается на текущую дату и время при создании команды
* creator: Поле типа ForeignKey, связывает с моделью User и обозначает создателя команды

1. UserProfile

* user: Поле типа OneToOneField, связывает с моделью User
* team: Поле типа ForeignKey, связывает с моделью Team

Через свойство profile добавляется расширение базовой модели пользователя Django (User), которое автоматически создает профиль пользователя, если он не существует.

Таким образом, разработаны модели django необходимые для взаимодействия с БД.

3.5.1.2 Представления

Представления в Django определяют логику обработки запросов от пользователей и формирования ответов. Они могут выполнять различные операции, такие как чтение и запись данных, валидация данных, рендеринг шаблонов и другие действия. Представления могут работать с данными из моделей, обрабатывать формы, вызывать другие представления и выполнять бизнес-логику приложения. Представления могут быть функциональными или классами, в зависимости от сложности логики обработки запросов. Модели обеспечивают структуру данных и бизнес-логику, а представления управляют отображением и обработкой данных. Представления используют модели для получения, обновления и сохранения данных в базе данных. Модели и представления работают совместно для обеспечения функциональности веб-приложения, где модели предоставляют доступ к данным, а представления обрабатывают запросы пользователей.

В ходе разработки сервиса были реализованы представления (views) в рамках Django REST Framework, которые реализуют различные аспекты API для работы с пользователями, фотографиями и командами.

UserViewSet – это представление реализует операции для модели User, используя viewsets.ModelViewSet. Оно позволяет создавать, читать, обновлять и удалять пользователей.

UploadPhotoView – представление, основанное на APIView, позволяет пользователям загружать фотографии. Оно принимает загруженные изображения, выполняет их сегментацию с помощью функции predict, сохраняет результат и связывает его с пользователем. Для этого используются сериализаторы UploadedPhotoSerializer и SegmentedPhotoSerializer. Для обработки файлов используется MultiPartParser и FormParser. Доступ к этому представлению имеют только аутентифицированные пользователи (IsAuthenticated), а аутентификация осуществляется через JWT (JWTAuthentication)

UserProfileView является классом, который обрабатывает запросы для получения профиля пользователя по его имени пользователя (username). В данном случае, метод get обрабатывает GET-запрос, который возвращает данные пользователя в формате JSON.

Класс UserDetail используется для операций чтения (получения) и обновления данных о пользователе, наследует от RetrieveUpdateAPIView, что позволяет получать информацию о пользователе (чтение) и обновлять эту информацию.

UserTeamView - класс, предназначенный для работы с командами пользователей. Он обрабатывает запросы GET, POST, PUT и DELETE для просмотра, добавления, обновления и удаления информации о командах, связанных с пользователем.

RegisterUser - класс, который обрабатывает POST-запрос для регистрации нового пользователя. Он получает данные о пользователе из запроса (username, password, firstname, lastname, email), создает нового пользователя в базе данных и возвращает соответствующий JSON-ответ.

update\_team\_photo обрабатывает POST-запрос для обновления фотографии команды. Функция получает идентификаторы фотографии (photo\_id) и команды (team\_id), а также флаг checked, указывающий на необходимость обновления фотографии. Функция ищет объекты команды (Team) и сегментированной фотографии (SegmentedPhoto) по их идентификаторам, создает или находит соответствующие объекты UploadedPhoto, UserPhoto и TeamPhoto и обновляет информацию о видимости фотографии для команды в зависимости от значения флага checked.

Класс CommentViewSet используется для работы с комментариями. Метод list получает список комментариев, относящихся к конкретной фотографии команды (teamphotoid), и возвращает их в формате JSON с помощью сериализатора. Метод `create` создает новый комментарий к фотографии команды, указанной teamphotoid. Запрашивает данные комментария из запроса, заменяет ID фотографии команды на объект TeamPhoto, валидирует данные с помощью CommentCreateSerializer и сохраняет комментарий в базу данных.

Класс UserPhotoDetail получает детальную информацию о конкретной пользовательской фотографии по её идентификатору.

Класс TeamView реализует методы post для создания новой команды и get для получения списка всех команд.

Класс TeamUsersView необходим для получения списка пользователей, принадлежащих определенной команде (team\_id).

3.5.1.3 Эндпоинты

Эндпоинты (endpoint) относятся к конечным точкам API, к которым клиенты могут обращаться для выполнения определенных операций или получения определенных данных. Эндпоинты в API определяют доступные операции, такие как GET, POST, PUT, DELETE. Эндпоинты описываются в Swagger с помощью специального формата, который содержит информацию о доступных эндпоинтах, методах запроса, параметрах, структуре данных и других деталях API.

Swagger — это набор инструментов для проектирования, создания, документирования и потребления веб-сервисов RESTful API. Он позволяет разработчикам описывать структуру и функционал API с помощью JSON или YAML файлов.

Все эндпоинты проетка описаны в Swagger, расположенном в папке проекта.

3.5.1.4 База данных

База данных (БД) — это организованное собрание данных. Она представляет собой структурированную коллекцию данных, которая облегчает хранение, обновление, поиск и управление информацией.

Бэкенд использует базу данных для хранения, организации и извлечения данных, необходимых для работы приложения. Бэкенд может выполнять запросы к базе данных для получения информации, добавления новых данных, обновления существующих записей или удаления данных в соответствии с бизнес-логикой приложения.

На рисунке 13 представлена схема базы данных сервиса.

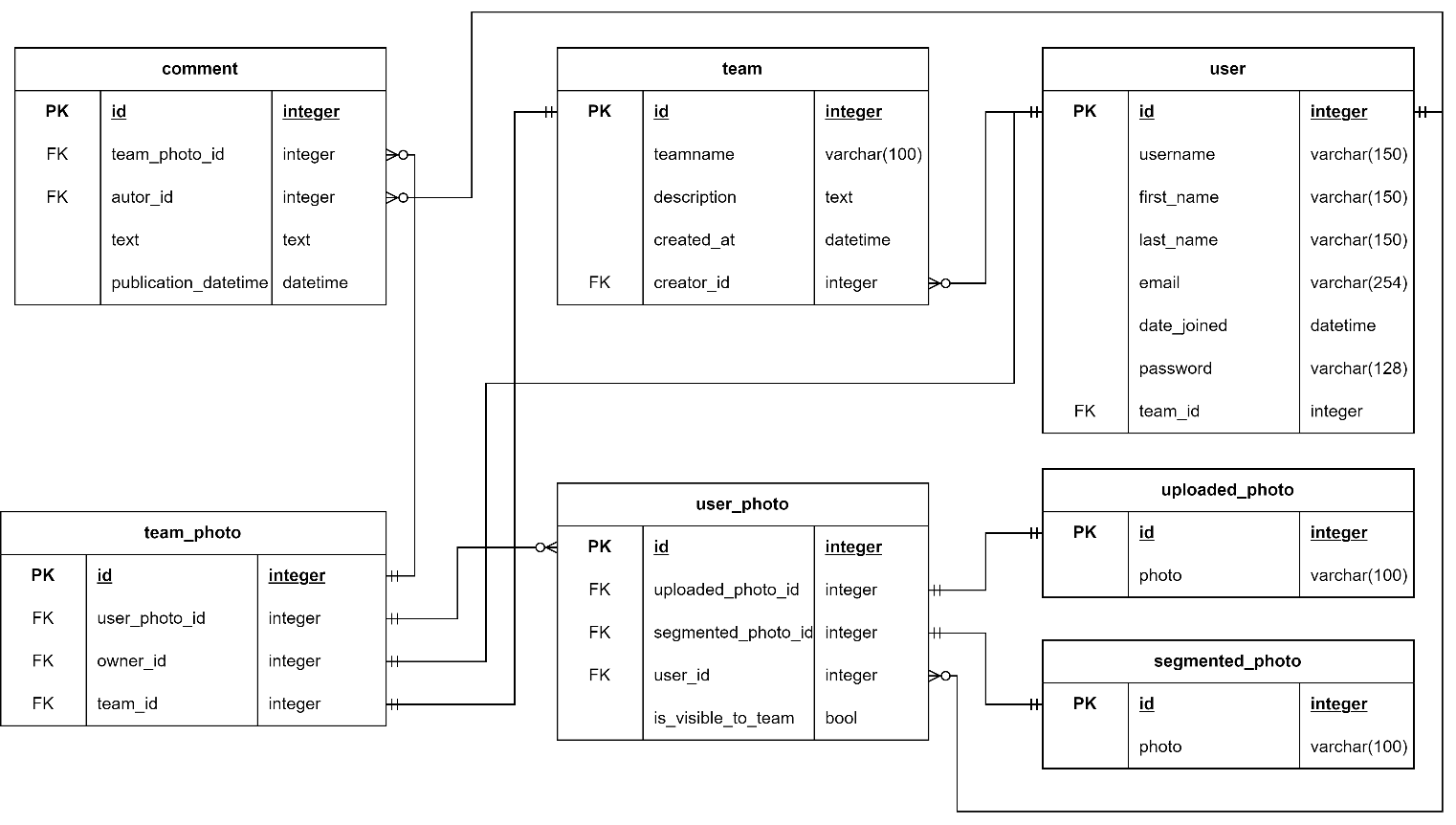


Рисунок 13 – Схема базы данных

3.5.2 Frontend

Фронт веб-приложения отвечает за то, как пользователи видят и взаимодействуют с интерфейсом. Он представляет собой то, что пользователь видит на экране и как он взаимодействует с элементами интерфейса.

Маршруты веб-приложения определяют, какой компонент должен быть отображен при определенном URL-адресе. Когда пользователь переходит по определенному URL, маршрутизатор определяет, какой компонент должен быть загружен и отображен на экране.

Компоненты веб-приложения представляют собой отдельные части интерфейса, которые могут быть загружены и отображены на экране. Они содержат логику и представление, необходимые для работы с определенными данными или функциональностью.

Маршруты определяют, какие компоненты должны быть загружены и отображены на экране в зависимости от URL-адреса, который пользователь посещает.

3.5.2.1 Маршруты

Маршруты связывают веб-адреса с определенными страницами и другими компонентами, используя мощный механизм рендеринга React и условную логику для программного включения и выключения маршрутов [11].

В разрабатываемом веб-сервисе определены маршруты для навигации пользователей по разным разделам приложения.

При входе на главную страницу по умолчанию пользователи попадают на адрес “/”. На этой странице пользователи могут увидеть основное содержимое приложения, приветственное сообщение.

“/photos” – страница “Фотографии” на которой пользователи могут просматривать, загружать, сегментировать свои изображения.

Адрес “/login” содержит страницу входа. Здесь пользователи могут войти в систему, введя свои учетные данные.

Страница регистрации расположена по адресу “/register”. На этой странице пользователи могут зарегистрировать новую учетную запись в сервисе.

Завершить сеанс работы и выйти из учетной записи с сервисом пользователь может, перейдя на страницу выхода по адресу “/logout”.

На странице “/profile” пользователи могут управлять своей учетной записью и просматривать персонализированную информацию.

“/team” – адрес, по которому расположена страница команд, на которой пользователи могут работать с командой в сервисе и управлять участниками.

“/photos/:id” – страница, на которой расположено оригинальное фото и его сегментированная версия для ознакомления пользователя с результатами сегментации.

“/userguide” – руководство пользователя, для ознакомления пользователя с правилами работы с веб-сервисом.

3.5.2.2 Компоненты

Компоненты позволяют разбить интерфейс на независимые части, про которые легко думать в отдельности. Их можно складывать вместе и использовать несколько раз [12].

**Main**

В компоненте Main отображается информация о сервисе, его целях, технологиях и пользователям предлагается присоединиться к использованию сервиса. Также предоставляется возможность перейти к тестированию сервиса или ознакомиться с руководством пользователя.

**Photos**

Photos отвечает за отображение галереи изображений пользователя и обработку действий пользователя, таких как выбор и загрузка новых изображений.

При загрузке страницы компонент Photos инициализирует состояния, такие как selectedFile, uploadedPhotos, selectedPhotos и isCheckedArray, а также получает userId из локального хранилища. Затем выполняется проверка наличия токена доступа, чтобы обеспечить аутентификацию пользователя.

При выборе файла пользователем срабатывает функция handleFileSelect, которая сохраняет выбранный файл в состоянии selectedFile. Затем пользователь может отправить выбранный файл на сервер с помощью функции handleUploadFile, которая создает FormData объект с выбранным файлом и отправляет POST запрос на сервер для загрузки изображения.

Для отображения галереи изображений используется условное отображение на основе наличия загруженных фотографий. Если у пользователя есть загруженные фотографии, то они отображаются в карточках с возможностью выбора для добавления на страницу команды. При изменении состояния чекбокса вызывается функция handleCheckboxChange, которая обрабатывает выбор или отмену выбора изображения для добавления на страницу команды.

Также в коде есть функции для обновления данных на сервере, такие как updateTeamPhoto, которая отправляет POST запрос для обновления информации о выбранных изображениях, и getTeamId, которая асинхронно получает teamId пользователя для дальнейших операций.

Компонент Photos обеспечивает пользователю возможность загрузки, просмотра и выбора изображений, а также взаимодействие с сервером для обновления данных.

**Login**

Компонент Login отвечает за процесс аутентификации пользователей в системе. При загрузке страницы происходит проверка наличия токена доступа в локальном хранилище: если токен уже есть, пользователь видит сообщение о том что вход в систему уже осуществлен, в противном случае отображается форма для ввода данных.

Форма входа содержит поля для ввода имени пользователя и пароля, которые связаны с соответствующими состояниями. При отправке формы вызывается функция submitHandler, которая отправляет данные на сервер для проверки и аутентификации.

Если аутентификация проходит успешно, полученные токены доступа сохраняются в локальном хранилище, а также запрашиваются и сохраняются данные пользователя. После успешного входа происходит перенаправление на главную страницу.

Компонент также содержит логику для обновления токена доступа, если это необходимо, и обработку возможных ошибок при запросах на сервер или во время аутентификации.

**Register**

Компонент Register отвечает за процесс регистрации новых пользователей в системе. При загрузке страницы пользователю предоставляется форма для ввода данных, таких как имя пользователя, пароль, имя, фамилия и электронная почта. Введенные пользователем данные отправляются на сервер для обработки и регистрации.

В компоненте используются хуки useState для управления состояниями, такими как загрузка данных (loading), данные формы ввода и сообщения об ошибках. Пользователю предоставляется возможность зарегистрироваться, а также перейти на страницу входа, если у него уже есть аккаунт.

При успешной регистрации нового пользователя данные отправляются на сервер с использованием метода POST. В случае возникновения ошибок, таких как логин, который уже занят, или других проблем, выводятся соответствующие сообщения об ошибке. После завершения процесса регистрации пользователь перенаправляется на главную страницу.

**Logout**

Компонент содержит функцию handleLogout, которая представляет собой процесс выхода пользователя из системы. При вызове этой функции происходит удаление данных о пользователе из локального хранилища, таких как токены доступа, имя пользователя, идентификатор пользователя, фамилия и имя. После этого выполняется функция onLogout, которая может содержать дополнительные действия при выходе пользователя.

Этот процесс выхода пользователя важен для обеспечения безопасности и защиты данных.

**Account**

Компонент Account в React отвечает за отображение информации о пользователе, его редактирование и сохранение изменений. При загрузке компонента он проверяет, аутентифицирован ли пользователь, и если да, загружает данные о пользователе из API. В случае успешной загрузки данные о пользователе отображаются, и пользователь может нажать кнопку редактирования, чтобы изменить свои данные.

При нажатии на кнопку редактирования открывается режим редактирования, где пользователь может изменить своё имя, фамилию и email. После внесения изменений и нажатия кнопки "Сохранить", данные отправляются на сервер для обновления. В случае успешного обновления информация о пользователе обновляется на странице.

Если токен доступа истек или недействителен, компонент автоматически пытается обновить токен через запрос на сервер. После успешного обновления токена, данные пользователя обновляются, и пользователь может продолжить взаимодействие с приложением.

Компонент Account содержит логику для управления состоянием редактирования, хранения измененных данных пользователя, обновления токена доступа и отправки запросов к API для загрузки и обновления данных пользователя. Все это делает компонент удобным для пользователей, позволяя им управлять своим профилем в приложении.

Компонент Account взаимодействует с несколькими конечными точками (эндпоинтами) для работы с информацией о пользователе:

1. Загрузка данных пользователя – при загрузке компонента Account, он отправляет GET запрос на эндпоинт /api/user/ для получения информации о текущем пользователе. В запросе передается токен доступа из localStorage для аутентификации. При успешном получении данных информация о пользователе отображается на странице.
2. Обновление данных пользователя – при нажатии кнопки "Сохранить" в режиме редактирования, компонент отправляет PUT запрос на эндпоинт /api/user/ с обновленными данными пользователя (имя, фамилия, email) для сохранения изменений. В запросе также передается токен доступа для аутентификации. При успешном обновлении данных информация о пользователе обновляется на странице.

**Team**

Компонент Team играет роль контроллера состояния пользователя относительно его командной принадлежности. При получении пропса isAuthenticated он определяет, аутентифицирован ли пользователь. Затем, с помощью эффекта useEffect, он отправляет запрос на сервер для проверки состоит ли пользователь в команде, и на основе полученной информации обновляет свой внутренний стейт. В соответствии с этим стейтом, компонент решает, какую страницу отобразить: либо UserTeamPage, показывающую информацию о команде и ее участниках, либо NoTeamPage, предлагающую пользователю создать свою команду или ожидать приглашения.

UserTeamPage дает возможность просматривать данные о команде, участниках, фотографиях и комментариях. Пользователь может также приглашать других участников в команду, управлять своим участием в команде и оставлять комментарии к фотографиям.

NoTeamPage предоставляет пользователю интерфейс для создания новой команды, где можно указать название и описание команды.

Для взаимодействия с данными используются различные API эндпоинты, такие как получение информации о текущем пользователе, управление участием в команде, получение списка участников и фотографий команды, а также отправка и получение комментариев.

**PhotoDetail**

Данный компонент отвечает за отображение детальной информации о конкретной фотографии, включая загруженное фото и его сегментированную версию.

В компоненте используются хуки useState и useEffect для управления состоянием компонента и выполнения асинхронных запросов к API.

При рендеринге компонента сначала извлекается параметр id из URL с помощью хука useParams. Затем, с помощью useEffect, выполняется запрос к API для получения данных по конкретной фотографии. Полученные данные сохраняются в состоянии с помощью функции setPhotoData.

После получения данных компонент отображает информацию о фотографии, включая само изображение загруженной и сегментированной фотографии. Также компонент включает кнопку для возврата к странице с галереей фотографий.

Важным элементом компонента является компонент Legend, который представляет легенду для цветов, используемых в сегментированной фотографии. Цвета соответствуют определенным категориям (вода, земля, дорога, здание, растение) и отображаются с соответствующими названиями.

Таким образом, компонент PhotoDetail обеспечивает пользователю возможность просмотра детальной информации о фотографии, включая сравнение загруженной и сегментированной версии, а также ознакомление с легендой цветовых категорий.

**UserGuide**

Компонент UserGuide представляет собой страницу с руководством пользователя для веб-приложения. На этой странице пользователю предоставляются основные шаги для использования приложения, инструкции по началу работы, ключевые функции приложения, а также советы по помощи и поддержке. Компонент содержит различные разделы с информацией и ссылками для удобства пользователя.

Ключевые особенности компонента UserGuide: отображение текстового руководства для пользователей приложения, предоставление инструкций по регистрации, входу в аккаунт и использованию основных функций приложения, редставление списка основных функций приложения с примерами ссылок для навигации, ссылки на разделы приложения, такие как галерея фотографий, страница команды и личный кабинет пользователя, возможность перейти на GitHub репозиторий проекта для просмотра исходного кода.

ЗАКЛЮЧЕНИЕ

В ходе исследования были рассмотрены различные методы сегментации объектов на изображениях, изучены архитектуры нейронных сетей, применяемых для этой задачи, и разработана эффективная модель сегментации объектов. Обучение модели с использованием выбранных методов позволило достичь высокой точности распознавания объектов на спутниковых снимках. Выбранные метрики оценки качества модели подтвердили ее надежность и точность.

Также были изучены различные архитектуры нейронных сетей, применяемых для сегментации спутниковых снимков. В работе были рассмотрены такие архитектуры, как U-Net, LinkNet, DeepLabV3Plus. Каждая из этих архитектур имеет свои особенности и преимущества в контексте сегментации спутниковых снимков.

Особое внимание было уделено разработке пользовательского интерфейса веб-сервиса, обеспечивающего удобство работы с моделью сегментации объектов. Этот инструмент позволит пользователям легко загружать изображения, выполнять сегментацию объектов и получать результаты в удобном формате. В работе были рассмотрены различные типы задач сегментации спутниковых снимков. Каждая из этих задач требует специфических методов и подходов для достижения точной и качественной сегментации

В заключение сегментация спутниковых снимков является важным инструментом для анализа и интерпретации данных, полученных с помощью спутниковых систем. Выбор подходящего метода и архитектуры для сегментации зависит от конкретной задачи, типа спутниковых снимков и требований к производительности. Понимание различных методов и архитектур помогает в выборе оптимального подхода и достижении высокой точности и качества сегментации спутниковых снимков.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Постников В.М., Спиридонов С.Б. Методы выбора весовых коэффицентов локальных критериев // Наука и образование. - М.: МГТУ им. Н.Э.Баумана, 2015. - С. 267-287.
2. Постников В.М. Основы эксплуатации АСОИнУ. Часть 2. Администрирование и развитие: учеб, пособие. М.: Изд- во МГТУ им. Н.Э. Баумана, 2015. 190 с.
3. Лукашик Д.В. Анализ современных методов сегментации изображений [Текст] / Лукашик Д.В. // ЭКОНОМИКА И КАЧЕСТВО СИСТЕМ СВЯЗИ. — 2022. — № 2. — С. 57-65.
4. Что такое семантическая сегментация? // Training Data URL: https://trainingdata.ru/markup-dark-side-ml/tpost/x2lu8a64o1-chto-takoe-semanticheskaya-segmentatsiya (дата обращения: 20.05.2024)
5. Глек П. U-Net: нейросеть для сегментации изображений / Павел Глек [Электронный ресурс] // Neurohive : [сайт]. — URL: https://neurohive.io/ru/vidy-nejrosetej/u-net-image-segmentation/ (дата обращения: 20.12.2023)
6. U-Net Architecture Explained / [Электронный ресурс] // GeeksForGeeks : [сайт]. — URL: https://www.geeksforgeeks.org/u-net-architecture-explained/ (дата обращения: 20.12.2023).
7. DeepLabv3 // Medium URL: https://medium.com/@itberrios6/deeplabv3-c0c8c93d25a4 (дата обращения: 21.05.2024).
8. Natarajan V.A., Kumar M.S., Patan R., Kallam S., Mohamed M.Y.N. Segmentation of Nuclei in Histopathology images using Fully Convolutional Deep Neural Architecture // 2020 International Conference on Computing and Information Technology. - Tabuk: 2020. - С. 319-325.
9. Django Documentation // Django URL: https://docs.djangoproject.com/en/5.0/ (дата обращения: 27.04.2024).
10. SQLite Documentation // SQLite URL: https://www.sqlite.org/docs.html (дата обращения: 27.04.2024).
11. Маршрутизация в React Router: как она работает и почему ее выбирают разработчики // Хекслет URL: https://ru.hexlet.io/blog/posts/react-router-v6#nastroyka-vlozhennyh-marshrutov-v-react-router-v6 (дата обращения: 20.05.2024).
12. Компоненты и пропсы // React URL: https://ru.legacy.reactjs.org/docs/components-and-props.html (дата обращения: 20.05.2024).

ПРИЛОЖЕНИЕ A ТЕХНИЧЕСКОЕ ЗАДАНИЕ

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления»

|  |  |  |
| --- | --- | --- |
| Утверждаю  Заведующий кафедрой ИУ-5 |  | Согласовано  научный руководитель |
| \_\_\_\_\_\_\_\_\_\_\_В.И.Терехов  "\_\_"\_\_\_\_\_\_\_\_\_\_2024 г. |  | \_\_\_\_\_\_\_\_\_\_А.Н. Нардид  "\_\_"\_\_\_\_\_\_\_\_\_\_2024 г. |

**Сегментация объектов на спутниковых снимках**

Техническое задание

(вид документа)

писчая бумага

(вид носителя)

5

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Шимолина Полина Кирилловна |
| "\_\_"\_\_\_\_\_\_\_\_\_\_\_2024 г. |  |

Москва - 2024

1. Введение

Сервис сегментации объектов на спутниковых снимках. Предназначен для автоматического выделения различных объектов на изображениях.

1. Основания для разработки

Основанием для разработки является задание на выпускную квалификационную работу, подписанное руководителем выпускной работы и утверждённое заведующим кафедрой ИУ5 МГТУ им. Н.Э. Баумана 15 декабря 2023 года.

1. Назначение разработки

Назначением работы является создание сервиса, позволяющего автоматизировать выделение различных объектов на изображениях с целью улучшения анализа и интерпретации данных, полученных с помощью спутниковых снимков.

1. Требования к программе или программному изделию
   1. Требования к функциональным характеристикам:

Программа должна выполнять следующие функции:

* + 1. Регистрация
    2. Вход в аккаунт
    3. Запрет доступа неавторизованным пользователям
    4. Запрет на регистрацию аккаунта с существующим именем пользователя
    5. Загрузка изображения
    6. Получение сегментированного изображения
    7. Просмотр галереи
    8. Просмотр сегментированного фото
    9. Создание команды
    10. Добавление в команду нового участника
    11. Выход из команды
    12. Добавление изображения на страницу команды
    13. Удаление изображения со страницы команды
    14. Создание комментария
    15. Просмотр личного кабинета
    16. Изменение данных в личном кабинете
    17. Отмена изменений данных в личном кабинете
    18. Просмотр руководства пользователя
    19. Выход из аккаунта
    20. Сохранение изображения на устройство
  1. Требования к составу и параметрам технических средств;

Минимальные системные требования для работы веб-сервиса:

* + 1. Подключение к интернету
    2. Установленный веб-браузер
    3. Компьютерная мышь
  1. Требования к информационной и программной совместимости
     1. Требования к исходным кодам и языкам программирования
        1. Python 3.11 для написания модели
        2. Python Django для backend
        3. JavaScript для frontend
        4. SQLite для хранения данных
        5. Библиотеки для создания модели и обработки данных в Python

1. Требования к программной документации

Для демонстрации разрабатываются следующие документы:

1. Техническое задание
2. Программа и методика испытаний
3. Руководство оператора
4. Графический материал по проекту в формате презентации
5. Технико-экономические показатели

Требования к данному разделу не предъявляются.

1. Стадии и этапы разработки
   1. Этапы разработки

График выполнения отдельных этапов работ приведён в таблице 1 соответствии с приказом об организации учебного процесса в 2023/20234 учебном году.

Таблица 1 – Этапы разработки

| **№ п/п** | **Наименование этапа и содержание работ** | **Сроки исполнения** |
| --- | --- | --- |
| 1 | Постановка задач разработки, анализ инструментов и программного обеспечения для создания приложения | Ноябрь-декабрь 2023 г. |
| 2 | Формулирование проблемы, цели и задач работы | Январь 2024 г. |
| 3 | Разработка модели для сегментации и веб-сервиса | Февраль-март 2024 г. |
| 4 | Тестирование и отладка модели и сервиса | Март-апрель 2024 г. |
| 5 | Оформление документации | Апрель 2024 г. |
| 6 | Демонстрация научному руководителю итоговой работы | Май 2024 г. |

* 1. Перечень разрабатываемых программных документов

График разработки отдельных программных документов приведён в таблице 2 соответствии с утвержденным календарным планом для выпускной квалификационной работы.

Таблица 2 – График разработки отдельных программных документов

| **№ п/п** | **Наименование этапов разработки отдельных программных документов** | **Сроки выполнения этапов** | **Принимающее лицо** | |
| --- | --- | --- | --- | --- |
| **Должность** | **подпись, ФИО** |
|  | Разработка технического задания | 10.03.2024 | Руководитель ВКР | А.Н.Нардид |
|  | Разработка программы и методики испытания | 01.04.2024 | Руководитель ВКР | А.Н.Нардид |
|  | Подготовка доклада и презентации | 25.05.2024 | Руководитель ВКР | А.Н.Нардид |
|  | Получение заключения научного руководителя | 28.05.2024 | Руководитель ВКР | А.Н.Нардид |

1. Порядок контроля и приёмки

Приём программного изделия в виде испытаний осуществляется в ходе «Защиты макетов программ – предварительной защиты ВКРБ» в период с 15 по 24 мая 2024 года в соответствие с разработанной программой и методикой испытаний.

ПРИЛОЖЕНИЕ B ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления»

|  |  |  |
| --- | --- | --- |
| Утверждаю  Заведующий кафедрой ИУ-5 |  | Согласовано  научный руководитель |
| \_\_\_\_\_\_\_\_\_\_\_В.И.Терехов  "\_\_"\_\_\_\_\_\_\_\_\_\_2024 г. |  | \_\_\_\_\_\_\_\_\_\_А.Н. Нардид  "\_\_"\_\_\_\_\_\_\_\_\_\_2024 г. |

**Сегментация объектов на спутниковых снимках**

ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

(вид документа)

писчая бумага

(вид носителя)

8

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Шимолина Полина Кирилловна |
| "\_\_"\_\_\_\_\_\_\_\_\_\_\_2024 г. |  |

Москва – 2024

1. **Объект испытаний**

Веб-приложение для выделения объектов на спутниковых снимках с использованием модели машинного обучения.

1. **Цель испытаний**

Цель испытания – проверка функционирования всех указанных в техническом задании функций программы.

1. **Состав предъявляемой документации**

На испытания программного продукта предъявляются следующие документы:

1. Техническое задание
2. Программа и методика испытаний
3. Руководство оператора
4. **Технические требования**

Требования к программной документации

Комплектность программной документации должна удовлетворять разделу данного документа "Состав предъявляемой документации".

* 1. **Требования к техническим характеристикам**
     1. **Требования к составу аппаратного обеспечения**

Сервис должен выполняться на сервере со следующими характеристиками:

* CPU: 1 vCPU (3GHz+)
* Оперативная память: 1 GB DDR4 ECC
* SSD: 20 GB
* Пропускная способность 100 Мбит/с
  + 1. **Требования к составу программного обеспечения**

Для доступа к данному веб-приложению необходимо наличие браузера.

1. **Порядок проведения испытаний**

Испытания системы будут проводиться в следующем порядке:

* + - 1. Регистрация или вход в аккаунт
      2. Добавление команды
      3. Взаимодействие с сервисом

1. **Методы испытаний**

| **№** | **№ пункта ТЗ** | **Выполняемые действия** | **Результат** |
| --- | --- | --- | --- |
| 1. | 4.1.1 Регистрация | Пользователь:   1. Открывает приложение 2. Нажимает кнопку «Зарегистрироваться». 3. Вводит имя пользователя, email, пароль, имя и фамилию 4. Нажимает кнопку «Зарегистрироваться» | Регистрируется аккаунт пользователя, на экране появляется форма входа |
| 2. | 4.1.2 Вход в аккаунт | Пользователь:   1. Открывает приложение 2. Нажимает кнопку «Войти». 3. Вводит имя, пароль 4. Нажимает кнопку «Войти» | На навигационной панели отображаются кнопки «Попробовать», «Моя команда», имя пользователя и «Выйти» |
| 3. | 4.1.3 Запрет доступа неавторизованным пользователям | Пользователь:   1. Открывает приложение 2. Нажимает кнопку «Войти». 3. Вводит имя пользователя, которого не существует или неверный пароль 4. Нажимает кнопку «Войти» | Отображается уведомление «Что-то пошло не так, возможно пользователя с таким именем не существует или пароль введет не верно.» |
| 4. | 4.1.4 Запрет на регистрацию аккаунта с существующим именем пользователя | Пользователь:   1. Открывает приложение 2. Нажимает кнопку «Зарегистрироваться». 3. Вводит имя пользователя, на которое уже зарегистрирован аккаунт 4. Нажимает кнопку «Зарегистрироваться» | Отображается уведомление «Что-то пошло не так. Возможно такое имя пользователя уже занято.» |
| 5. | 4.1.5 Загрузка изображения | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Попробовать» 3. Нажимает кнопку «Choose File/Выберите файл» 4. Выбирает файл 5. Нажимает кнопку «Open/Открыть» | Выбранное пользователем фото отображается на странице. |
| 6. | 4.1.6 Получение сегментированного изображения | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Попробовать» 3. Загружает изображение 4. Нажимает кнопку «Отправить файл на сервер». | Сегментированное фото отображается в галерее. |
| 7. | 4.1.7 Просмотр галереи | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Попробовать» | В разделе «Моя галерея» отображаются все ранее добавленные изображения. |
| 8. | 4.1.8 Просмотр сегментированного фото | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Попробовать» 3. Нажимает на карточку загруженного ранее изображения | Отображаются 2 изображения: загруженное и сегментированное, и легенда. |
| 9. | 4.1.9 Создание команды | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Моя команда» 3. Вводит название команды в поле «Введите название команды» 4. Вводит описание команды в поле «Введите описание команды» 5. Нажимает кнопку «Создать команду» | Отображается страница с названием команды, списком участников и галереей изображений. |
| 10. | 4.1.10 Добавление в команду нового участника | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Моя команда» 3. Вводит имя пользователя, которого хочет добавить в команду в поле «Введите юзернейм» 4. Нажимает кнопку «Пригласить в команду» | Добавленный пользователь появляется в списке команды. |
| 11. | 4.1.11 Выход из команды | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Моя команда» 3. Нажимает кнопку «Выйти из команды» | Отображается страница с надписью «Вы не состоите в команде». |
| 12. | 4.1.12 Добавление изображения на страницу команды | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Попробовать» 3. Ставит галочку «Добавить на страницу команды» под изображением, которое хочет добавить | Под добавленным на страницу команды изображением стоит галочка. На странице «Моя команда» отображается добавленное изображение. |
| 13. | 4.1.13 Удаление изображения со страницы команды | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Попробовать» 3. Ставит галочку «Добавить на страницу команды» под изображением, которое хочет добавить | Под удаленным со страницы команды изображением не стоит галочка. На странице «Моя команда» не отображается удаленное изображение. |
| 14. | 4.1.14 Создание комментария | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Моя команда» 3. Пишет комментарий под изображением в поле «Add comment» 4. Нажимает кнопку с изображением бумажного самолетика | Под изображением появляется комментарий с надписью информирующей о времени отправки и автором комментария. |
| 15. | 4.1.15 Просмотр личного кабинета | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку со своим именем пользователя | Отображаются данные аккаунта. |
| 16. | 4.1.16 Изменение данных в личном кабинете | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку со своим именем пользователя 3. Нажимает кнопку в виде карандаша 4. Вносит изменения в имя, фамилию или почту 5. Нажимает кнопку «Сохранить» | Данные аккаунта изменены, на странице отображаются измененные данные. |
| 17. | 4.1.17 Отмена изменений данных в личном кабинете | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку со своим именем пользователя 3. Нажимает кнопку в виде карандаша 4. Вносит изменения в имя, фамилию или почту 5. Нажимает кнопку «Отмена» | Данные аккаунта не изменены, на странице отображаются первоначальные данные. |
| 18. | 4.1.18 Просмотр руководства пользователя | Пользователь:   1. Открывает приложение 2. Нажимает кнопку «Руководство пользователя» | Отображается руководство пользователя. |
| 19 | 4.1.19 Выход из аккаунта | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Выйти» | Пользователь перенаправлен на страницу входа, на верхней панели появляются кнопки «Войти», «Зарегистрироваться». |
| 20 | 4.1.20 Сохранение изображения на устройство | Авторизованный пользователь:   1. Открывает приложение 2. Нажимает кнопку «Попробовать» 3. Нажимает на карточку изображения, которое хочет сохранить 4. Нажимает кнопку «Сохранить изображение» 5. Выбирает расположение на устройстве 6. Нажимает «Сохранить» | Изображение сохранено на устройстве по выбранному пути. |

1. **Результат испытаний**

Основой испытаний является демонстрация работы основных функций приложения: авторизация, работа в команде, получение сегментированного фото.

Испытание считается пройденным успешно, если в процессе демонстрации пользователь смог успешно авторизоваться или зарегистрироваться, загрузить фото и получить сегментированное фото, создать команду, добавить в нее изображения, добавить в нее участников, выйти из команды.

ПРИЛОЖЕНИЕ C РУКОВОДСТВО ОПЕРАТОРА

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления»

|  |  |  |
| --- | --- | --- |
| Утверждаю  Заведующий кафедрой ИУ-5 |  | Согласовано  научный руководитель |
| \_\_\_\_\_\_\_\_\_\_\_В.И.Терехов  "\_\_"\_\_\_\_\_\_\_\_\_\_2024 г. |  | \_\_\_\_\_\_\_\_\_\_А.Н. Нардид  "\_\_"\_\_\_\_\_\_\_\_\_\_2024 г. |

**Сегментация объектов на спутниковых снимках**

РУКОВОДСТВО ОПЕРАТОРА

(вид документа)

писчая бумага

(вид носителя)

21

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Шимолина Полина Кирилловна |
| "\_\_"\_\_\_\_\_\_\_\_\_\_\_2024 г. |  |

Москва – 2024

АННОТАЦИЯ

Сервис сегментации объектов на спутниковых снимках – это система, обеспечивающая сегментацию изображений, их хранение и совместный просмотр.

В настоящем документе приводится руководство оператора сервиса сегментации объектов на спутниковых снимках.

Документ предназначен для пользователей системы.

Настоящий документ содержит сведения о назначении системы, условиях ее выполнения, порядке действий оператора по работе с системой, сообщениях, выдаваемых оператору в процессе работы.

Документ выполнен в соответствии с требованиями государственного стандарта Единой системы программной документации ГОСТ 19.505-79 Руководство оператора. Требования к содержанию и оформлению.

СОДЕРЖАНИЕ

[1. НАЗНАЧЕНИЕ ПРОГРАММЫ 5](#_Toc167059761)

[1.1. Область применения 5](#_Toc167059762)

[1.2. Краткое описание возможностей 5](#_Toc167059763)

[1.3. Уровень подготовки пользователя 5](#_Toc167059764)

[2. УСЛОВИЯ ВЫПОЛНЕНИЯ ПРОГРАММЫ 6](#_Toc167059765)

[3. ВЫПОЛНЕНИЕ ПРОГРАММЫ 7](#_Toc167059766)

[3.1. Запуск приложения 7](#_Toc167059767)

[3.2. Регистрация 7](#_Toc167059768)

[3.3. Вход 8](#_Toc167059769)

[3.4. Выход 9](#_Toc167059770)

[3.5. Работа в личном кабинете 9](#_Toc167059771)

[3.5.1. Просмотр информации в личном кабинете 9](#_Toc167059772)

[3.5.2. Редактирование информации в личном кабинете 9](#_Toc167059773)

[3.6. Сегментация изображения 11](#_Toc167059774)

[3.6.1. Получение сегментированного изображения 11](#_Toc167059775)

[3.6.2. Просмотр личной галереи сегментированных изображений. 13](#_Toc167059776)

[3.6.3. Просмотр фото из галереи. 13](#_Toc167059777)

[3.7. Команда 14](#_Toc167059778)

[3.7.1. Создание команды 14](#_Toc167059779)

[3.7.2. Добавление участника в команду 15](#_Toc167059780)

[3.7.3. Добавление изображения на страницу команды 16](#_Toc167059781)

[3.7.4. Создание комментария под фото на странице команды 17](#_Toc167059782)

[3.7.5. Просмотр галереи команды 18](#_Toc167059783)

[3.7.6. Выход из команды 18](#_Toc167059784)

[3.8. Руководство пользователя 18](#_Toc167059785)

[3.8.1. Страница с руководством 19](#_Toc167059786)

[3.8.2. Подсказки 19](#_Toc167059787)

[4. СООБЩЕНИЯ ОПЕРАТОРУ 21](#_Toc167059788)

* + - 1. НАЗНАЧЕНИЕ ПРОГРАММЫ
  1. Область применения

Сервис сегментации объектов на спутниковых снимках находит широкое применение в различных областях. В геоинформационных системах (ГИС) он помогает автоматизировать обработку геопространственных данных, включая картографию, мониторинг изменений в окружающей среде и анализ земельного пользования.

В городском планировании и инфраструктуре он помогает анализировать застройку, оценивать состояние инфраструктуры и планировать развитие городской среды.

В экологии и охране окружающей среды сервис помогает мониторить экосистемы, определять зоны с особым природоохранного значения и отслеживать изменения в растительном покрове.

* 1. Краткое описание возможностей

Сервис обеспечивает выполнение следующих функций:

* Получение сегментированного изображения
* Хранение ранее сегментированных изображений
* Создание команд пользователей
* Совместный просмотр общих изображений
* Комментирование общих изображений
  1. Уровень подготовки пользователя

Система обладает интуитивно понятным интерфейсом позволяющим пользователям легко ориентироваться в разделах системы, быстро находить и усваивать нужную информацию, взаимодействовать в едином информационном пространстве сервиса.

1. УСЛОВИЯ ВЫПОЛНЕНИЯ ПРОГРАММЫ

Требования к общему программному обеспечению. Рабочее место поставщика данных, осуществляющего доступ к закрытой части портала, должно включать следующее программное обеспечение:

* Яндекс Браузер, Google Chrome или другой альтернативный браузер.

1. ВЫПОЛНЕНИЕ ПРОГРАММЫ
   1. Запуск приложения

Для запуска приложения в адресную строку необходимо ввести адрес веб-приложения, после чего откроется главная страница.

* 1. Регистрация

Переход на страницу регистрации осуществляется с главной страницы по нажатию на кнопку «Зарегистрироваться». На странице размещена форма регистрации.

Пользователю необходимо заполнить форму, включающую поля: “Username” – уникальное имя пользователя, “Email” – адрес электронной почты, “Password” – пароль, “First Name (optional)” – имя, “Last Name (optional)” - фамилия, среди которых “First Name (optional)” и “Last Name (optional)” – необязательные поля. После заполнения формы требуется нажать на кнопку «Зарегистрироваться».

На рисунке 1 показана страница регистрации.

Изображение выглядит как снимок экрана, текст, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рисунок 1 – Страница регистрации

Если регистрация успешна, пользователь перенаправляется на страницу входа.

* 1. Вход

Переход на страницу входа осуществляется с главной страницы по нажатию на кнопку «Войти» или в результате успешной регистрации. На странице размещена форма входа.

Пользователю необходимо заполнить форму, включающую поля “Username”, “Password”, и нажать на кнопку «Войти».

На рисунке 2 показана страница входа.

Изображение выглядит как снимок экрана, текст, дизайн

Автоматически созданное описание

Рисунок 2 – Страница входа

Если вход успешен, пользователь перенаправляется на главную страницу и на верхней панели появляются кнопки «Попробовать», «Моя команда», «Выйти», а также кнопка с именем пользователя. На рисунке 3 изображен результат успешного входа.

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 3 – Результат успешного входа

* 1. Выход

Выход осуществляется нажатием на кнопку «Выход» на верхней панели.

В результате выхода пользователь перенаправляется на страницу входа и на верхней панели появляются кнопки «Вход» и «Регистрация».

* 1. Работа в личном кабинете
     1. Просмотр информации в личном кабинете

Для просмотра информации в личном кабинете пользователю следует нажать на кнопку со своим именем пользователя на верхней панели. В результате пользователь окажется на странице, где может ознакомиться с информацией в личном кабинете.

На рисунке 4 изображена страница личного кабинета пользователя.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 4 – Личный кабинет пользователя

* + 1. Редактирование информации в личном кабинете
       1. Переход в режим редактирования

Для перехода в режим редактирования информации в личном профиле пользователю, находясь на странице личного кабинета, необходимо нажать на кнопку «Редактировать» в виде карандаша. В результате появится форма с полями для изменения электронной почты, имени и фамилии.

Страница, демонстрирующая режим редактирования личной информации, изображена на рисунке 5.

Изображение выглядит как текст, программное обеспечение, веб-страница, Шрифт

Автоматически созданное описание

Рисунок 5 - Режим редактирования личной информации

* + - 1. Внесение и сохранение изменений

Находясь на странице личного кабинета в режиме редактирования, пользователь может поменять свое имя, фамилию или электронную почту в соответствующих полях. После внесения изменений пользователь нажимает кнопку «Сохранить».

В результате этого действия внесенные изменения сохранятся и отобразятся на странице личного кабинета, режим редактирования отключится.

На рисунке 6 продемонстрировано отображение сохраненных изменений в имени пользователя.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6 – Внесенные изменения сохранены

* + - 1. Внесение и отмена изменений

Находясь на странице личного кабинета в режиме редактирования, пользователь может поменять свое имя, фамилию или электронную почту в соответствующих полях и отменить внесенные изменения при помощи нажатия кнопки «Отменить».

В результате этого действия внесенные изменения не сохранятся и на странице личного кабинета отобразятся старые данные пользователя, режим редактирования отключится.

* 1. Сегментация изображения
     1. Получение сегментированного изображения

Для получения сегментированного изображения пользователь должен перейти на соответствующую страницу при помощи кнопки «Попробовать».

На открывшейся странице будет доступна кнопка «Выберите файл», как изображено на рисунке 7. Пользователю следует нажать на кнопку, в появившемся окне выбрать нужный файл и нажать кнопку «Открыть».

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 7 – Страница сегментации

В результате выбора изображения, выбранное изображение отобразится под кнопкой «Выберите файл» как продемонстрировано на рисунке 8.

Изображение выглядит как текст, снимок экрана, линия

Автоматически созданное описание

Рисунок 8 – Загруженное фото

Далее пользователь нажимает кнопку «Отправить на сервер» и сегментированное изображение появляется в разделе «Моя галерея», как можно увидеть на рисунке 9.

Изображение выглядит как снимок экрана, текст, программное обеспечение

Автоматически созданное описание

Рисунок 9 – Получено сегментированное изображение.

* + 1. Просмотр личной галереи сегментированных изображений.

Для ознакомления со своей личной галереей пользователю необходимо перейти на страницу галереи при помощи кнопки «Попробовать». Перед пользователем отобразится страница, на которой в разделе «Моя галерея» представлены все равнее сегментированные изображения, как показано на рисунке 10.

Изображение выглядит как текст, снимок экрана, фиолетовый, диаграмма

Автоматически созданное описание

Рисунок 10 – Личная галерея сегментированных изображений

* + 1. Просмотр фото из галереи.

Для просмотра конкретного фото из галереи пользователю необходимо перейти на страницу с использованием кнопки «Попробовать», в разделе «Моя галерея» выбрать изображение и нажать на карточку с эти изображением. В результате пользователь окажется на странице, на которой отображается первоначальное загруженное фото, сегментированное изображение и легенда, как продемонстрировано на рисунке 11. Для того чтобы вернуться в галерею пользователю следует воспользоваться кнопкой «Назад к фотографиям».

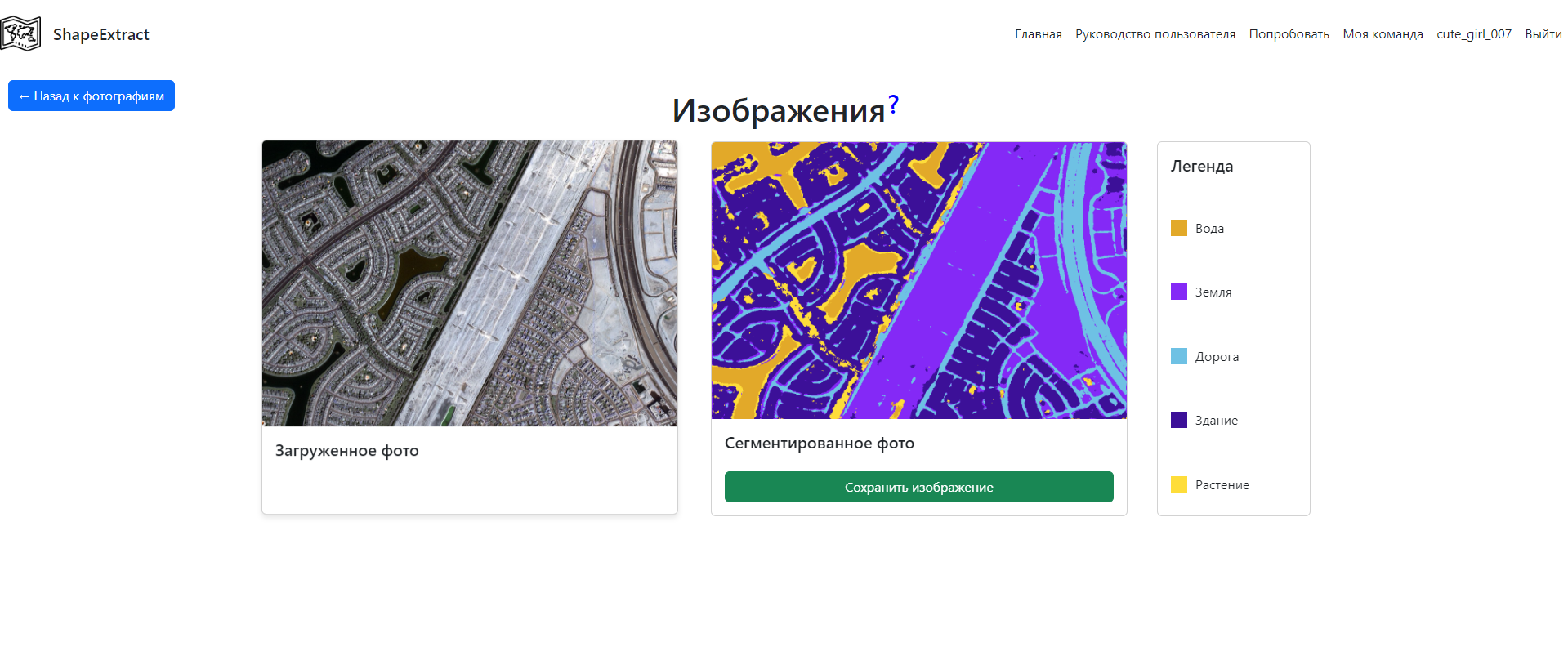


Рисунок 11 – Просмотр фото из галереи

* + 1. Сохранение фото на устройство

Для сохранения фото пользователю необходимо, находясь на странице нужного фото, нажать на кнопку «Сохранить изображение», в диалоговом окне выбрать местоположение на устройстве и нажать кнопку «Сохранить». В результате изображение появится по выбранному пути.

* 1. Команда
     1. Создание команды

Для создания команды пользователь переходит на страницу «Моя команда» и вводит название команды и описание в соответствующие поля как показано на рисунке 12.

Изображение выглядит как снимок экрана, текст, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 12 – Страница команды пользователя, с введенными данными команды

После этого пользователь нажимает кнопку «Создать команду». В результате создается команда и пользователь видит страницу своей команды, изображенную на рисунке 13.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 13 – Страница команды

* + 1. Добавление участника в команду

Для добавления нового участника в команду пользователю необходимо перейти на страницу команды по нажатию на кнопку «Моя команда», ввести имя пользователя этого участника и нажать кнопку «Пригласить в команду», как изображено на рисунке 14.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 14 – Введенное имя пользователя

В случае успеха пользователь появляется в списке команды (см. рис. 15).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 15 – Пользователь добавлен в команду

* + 1. Добавление изображения на страницу команды

Для добавления изображения на страницу команды пользователю надо попасть на страницу загрузки файлов при помощи кнопки «Попробовать». Затем необходимо проставить галочку под изображением, которое будет добавлено на страницу команды, как видно на рисунке 16.

Изображение выглядит как текст, снимок экрана, фиолетовый, дизайн

Автоматически созданное описание

Рисунок 16 – Галочка под изображением прожата

В результате на странице команды появится добавленное изображение в разделе «Наша галерея» (см. рис. 17).

Изображение выглядит как текст, снимок экрана, программное обеспечение, дизайн

Автоматически созданное описание

Рисунок 17 – Изображение на станицы команды

* + 1. Создание комментария под фото на странице команды

Для создания комментария пользователь переходит на страницу команды с помощью кнопки «Моя команда». Под изображением, под которым хочет оставить комментарий, пользователь пишет свой комментарий в поле «Add a comment» (см. рис. 18).

Изображение выглядит как текст, снимок экрана, программное обеспечение, дизайн

Автоматически созданное описание

Рисунок 18 – В поле комментария введен текст

После этого необходимо нажать на кнопку «Отправить» в виде бумажного самолетика. Комментарий появится под изображением с указанием автора, даты и времени отправки (см. рис. 19).

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рисунок 19 – Комментарий под изображением

* + 1. Просмотр галереи команды

Для просмотра галереи команды пользователь переходит на страницу команды с помощью кнопки «Моя команда». В результате галаерея будет отображаться в разделе «Наша галерея», как изображено на рисунке 20.

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рисунок 20 – Галерея команды

* + 1. Выход из команды

Чтобы осуществить выход из команды пользователю необходимо перейти на страницу команды с помощью кнопки «Моя команда», затем нажать на красную кнопку «Выйти из команды». После этого пользователь окажется на странице с предложением создать новую команду.

* 1. Руководство пользователя
     1. Страница с руководством

Для ознакомления с руководством пользователя пользователю следует нажать на кнопку «Руководство пользователя» на верхней панели. В результате отобразится руководство с вспомогательными гиперссылками как видно на рисунке 21.

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

Рисунок 21 – Руководство пользователя

* + 1. Подсказки

Подсказки – всплывающие окна, которые присутствуют на некоторых страницах системы. Чтобы ознакомиться с информацией на подсказке пользователю необходимо навести курсор на синий знак вопроса, изображенный на рисунке 22.

Изображение выглядит как текст, Шрифт, снимок экрана, Графика

Автоматически созданное описание

Рисунок 22 – Знак вопроса

После наведения пользователь увидит подсказку, пример которой изображен на рисунке 23.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 23 - Подсказка

1. СООБЩЕНИЯ ОПЕРАТОРУ

|  |  |
| --- | --- |
| Сообщение | Назначение |
| Что-то пошло не так, возможно пользователя с таким именем не существует или пароль введет не верно | Информирует пользователя о том, что пароль введен не верно или пользователь с введенным именем пользователя не зарегистрирован в системе |
| Заполните это поле | Информирует о том, что для осуществления входа или регистрации следует заполнить необходимое поле |
| Что-то пошло не так. Возможно такое имя пользователя уже занято | Информирует пользователя о том, что невозможно зарегистрировать аккаунт на введенное имя пользователя |
| Пользователь уже состоит в команде | Информирует о невозможности добавить данного пользователя в команду, так как он уже состоит в команде |