Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Институт информационных технологий

Факультет компьютерных технологий

Кафедра информационных систем и технологий

|  |
| --- |
| *К защите допустить*: |
| Заведующий кафедрой ИСиТ |
| \_\_\_\_\_\_\_\_\_\_\_\_А.И. Парамонов |

Пояснительнаязаписка

к дипломному проекту

на тему

**ВЕБ-СЕРВИС РАСПОЗНАВАНИЯ ГОСУДАРСТВЕННЫХ НОМЕРОВ ТРАНСПОРТНЫХ СРЕДСТВ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ OPENCV**

БГУИР ДП 1-40 01 01 01 148 ПЗ

|  |  |  |
| --- | --- | --- |
| Студент |  | В.В. Фролов |
| Руководитель |  | А.Л. Сицко |
| Консультанты: |  |  |
| *от кафедры ИСиТ* |  | А.Л. Сицко |
| *по экономической части* |  | В.Г. Горовой |
|  |  |  |
| Нормоконтролер |  | А.С. Шелягович |
|  |  |  |
| Рецензент |  |  |

Минск 2024

**Р** **Е** **Ф** **Е** **Р** **А** **Т**

ВЕБ-СЕРВИС РАСПОЗНАВАНИЯ ГОСУДАРСТВЕННЫХ НОМЕРОВ ТРАНСПОРТНЫХ СРЕДСТВ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ OPENCV: дипломный проект / В.В. Фролов. – Минск: БГУИР, 2024, – п.з. – 90 с., чертежей (плакатов) – 6 л. формата А1.

Объектом проектирования является программное средство для распознавания государственных регистрационных номеров транспортных средств с использованием библиотеки OpenCV.

Целью дипломного проекта является разработка веб-сервиса, который автоматизирует процесс распознавания регистрационных номеров, обеспечит удобное взаимодействие пользователей с системой и предоставит надежные инструменты для обработки и хранения данных.

Для достижения поставленной цели разработан программный комплекс, включающий серверное приложение, реализующее RESTful API, и клиентское веб-приложение с интуитивно понятным интерфейсом. Программное средство использует библиотеку OpenCV для обработки изображений и распознавания номерных знаков, а также реляционную базу данных Microsoft SQL Server для управления результатами и пользовательскими данными. Приложение реализовано на основе ASP.NET Core для серверной части и современных технологий фронтенд-разработки для клиентской части.

В процессе выполнения проекта был проведен анализ существующих решений и методов, использующих технологии компьютерного зрения и оптического распознавания символов (OCR). Разработаны функциональные модели системы, логическая и физическая модели базы данных, а также алгоритмы предобработки изображений, сегментации и распознавания символов. Реализация программного средства акцентирована на модульности системы, гибкости её расширения, производительности и защите данных.

Экономическое обоснование разработки показало её эффективность за счёт сокращения затрат на обработку данных вручную, повышения точности и скорости обработки информации, а также оптимизации процессов, связанных с регистрацией нарушений. Разработанный веб-сервис предоставляет удобный инструмент для автоматизации процессов идентификации транспортных средств, что значительно упрощает рабочие процессы и улучшает взаимодействие с конечными пользователями.

Созданное программное средство обеспечивает автоматизацию процессов распознавания регистрационных номеров, способствуя повышению производительности и точности обработки данных. Внедрение системы открывает возможности для её масштабирования и использования в различных сферах, таких как мониторинг дорожного движения, контроль доступа и управление парковками.

Министерство образования Республики Беларусь

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Учреждение образования БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ  Институт информационных технологий | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Факультет | КТ | | | Кафедра | | | ИСиТ | | | | | | | | | |
| Специальность | 1-40 01 01 | | | Специализация | | | | | | 01 | | | | | | |
| УТВЕРЖДАЮ | | | | | | | | | | | | | | | | |
|  | | | | | | | | |  | | | | А.И.Парамонов | | | |
| « | | | | | | | | |  | | » |  | | | 2024 | г. |
| ЗАДАНИЕ | | | | | | | | | | | | | | | | |
| по дипломному проекту студента | | | | | | | | | | | | | | | | |
| **Фролов Владимир Владимирович** | | | | | | | | | | | | | | | | |
| (фамилия, имя, отчество) | | | | | | | | | | | | | | | | |
| 1. Тема проекта: | | **Веб-сервис распознавания государственных номеров** | | | | | | | | | | | | | | |
| **Транспортных средств с использованием библиотеки OpenCV** | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| утверждена приказом по университету от | | | | « |  | » | |  | | | 2024 г. | | | № |  | |
| 2. Срок сдачи студентом законченной работы | | | | | 31 декабря 2024 года | | | | | | | | | | | |
| 3. Исходные данные к проекту | | | Тип операционной системы – ОС Windows 7 и выше; | | | | | | | | | | | | | |
| Язык программирования – C#; Среда .NET, ASP.NET, Razor Pages. клиент-серверная | | | | | | | | | | | | | | | | |
| архитектура. Обученная модель распознавания номеров. Перечень выполняемых | | | | | | | | | | | | | | | | |
| функций: регистрация, вход и выход пользователей, администрирование нарушений, | | | | | | | | | | | | | | | | |
| добавление и изменение нарушений | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| 4. Содержание пояснительной записки (перечень подлежащих разработке вопросов): | | | | | | | | | | | | | | | | |
| Введение | | | | | | | | | | | | | | | | |
| 1 Анализ предметной области | | | | | | | | | | | | | | | | |
| 2 Моделирование предметной области | | | | | | | | | | | | | | | | |
| 3 Проектирование и разработка программного средства | | | | | | | | | | | | | | | | |
| 4 Тестирование программного средства | | | | | | | | | | | | | | | | |
| 5 Руководство по эксплуатации программного средства | | | | | | | | | | | | | | | | |
| 6 Технико-экономическое обоснование разработки программного средства | | | | | | | | | | | | | | | | |
| Заключение | | | | | | | | | | | | | | | | |
| Список использованных источников | | | | | | | | | | | | | | | | |
| Приложение А. Исходный код программного средства | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |

|  |  |
| --- | --- |
| 5. Перечень графического материала (с точным указанием наименования) и обозначения | |
| вида и типа материала): |
| Диаграмма вариантов использования. Плакат – формат А1, лист 1. | |
| Диаграмма деятельности. Плакат – формат А1, лист 1. | |
| Диаграмма классов программного средства. Плакат – формат А1, лист 1. | |
| Добавление нарушения. Схема алгоритма – формат А1, лист 1. | |
| Авторизация. Схема алгоритма – формат А1, лист 1. | |
| Схема алгоритма поиска – формат А1, лист 1. | |
|  | |
| 6. Содержание задания по технико-экономическому обоснованию | |
| Расчет экономической эффективности от внедрения программного средства | |
|  | |
| Консультанты по дипломному проекту (с указанием разделов, по которому они консультируют):   |  | | --- | | Сицко А.Л. – разделы 1-5; | | Горовой В.Г. – раздел 6 (технико-экономическое обоснование); | | Шелягович А.С. – нормоконтроль. | | |

ПРИМЕРНЫЙ КАЛЕНДАРНЫЙ ГРАФИК ВЫПОЛНЕНИЯ ДИПЛОМНОГО ПРОЕКТА

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапов дипломного проекта | Объём готовности проекта в % | Срок выполнения этапа | Примечание |
| Первая опроцентовка (введение, разделы 1-3) | 30% | 16.09.24 – 25.10.24 | Консультант от кафедры |
| Вторая опроцентовка (разделы 3-5) | 60% | 26.10.24 – 30.11.24 | Руководитель проекта |
| Третья опроцентовка (разделы 5-6, заключение, список использованных источников) | 90% | 01.12.24 – 18.12.24 | Руководитель проекта |
| Консультации по оформлению графического материала и пояснительной записки, нормоконтроль | – | С 01.10.24  (согласно графику) | Нормоконтролёр |
| Итоговая проверка готовности дипломного проекта на заседании рабочей комиссии кафедры ИСиТ и допуск к защите в ГЭК | 100% | С 23.12.24  (согласно графику) | Председатель рабочей комиссии |
| Рецензирование дипломного проекта | 100% | С 26.12.24  (согласно распоряжению) | Рецензент |
| Защита дипломного проекта | 100% | С 18.01.25  (согласно приказу) | ГЭК |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Дата выдачи задания | 2024 г. | | Руководитель | |  | / А.Л.Сицко / |
|  |  | |  | |  |  |
| Задание принял к исполнению | |  | | / В.В. Фролов / | |

**СОДЕРЖАНИЕ**

[Введение 7](#_Toc185090219)

[1 Анализ предметной области 8](#_Toc185090220)

[1.1 Описание предметной области 8](#_Toc185090221)

[1.2 Обоснование актуальности разработки ПС 11](#_Toc185090222)

[1.3 Анализ методов, способов, подходов, методик, технологий 14](#_Toc185090223)

[1.4 Обзор существующих аналогов 16](#_Toc185090224)

[1.5 Выбор и обоснование средств разработки 19](#_Toc185090225)

[1.6 Спецификация требований 21](#_Toc185090226)

[2 Моделирование предметной области 24](#_Toc185090227)

[2.1 Разработка функциональных моделей предметной области 24](#_Toc185090228)

[2.2 Разработка и проектирование базы данных 26](#_Toc185090229)

[2.3 Разработка функциональной спецификации требований 28](#_Toc185090230)

[3 Проектирование и разработка программного средства 30](#_Toc185090231)

[3.1 Проектирование и разработка диаграммы вариантов использования 30](#_Toc185090232)

[3.2 Проектирование и разработка диаграммы деятельности 32](#_Toc185090233)

[3.3 Проектирование и разработка диаграммы классов 35](#_Toc185090234)

[3.4 Проектирование и разработка алгоритмов работы программы 38](#_Toc185090235)

[3.5 Проектирование и разработка базы данных 41](#_Toc185090236)

[4 Тестирование программного средства 43](#_Toc185090237)

[4.1 Выбор и обоснование видов тестирования 43](#_Toc185090238)

[4.2 Результаты тестирования 45](#_Toc185090239)

[4.3 Вывод тестирования 50](#_Toc185090240)

[5 Руководство по эксплуатации программного средства 51](#_Toc185090241)

[5.1 Назначение и условия применения 51](#_Toc185090242)

[5.2 Руководство пользователя 51](#_Toc185090243)

[6.2 Расчет инвестиций в разработку веб-приложения 60](#_Toc185090244)

[6.3 Расчет экономического эффекта от реализации программного средства на рынке 63](#_Toc185090245)

[6.4 Расчет показателей экономической эффективности разработки и реализации веб-сервиса 64](#_Toc185090246)

[Заключение 65](#_Toc185090247)

[Список использованных источников 66](#_Toc185090248)

[Приложение А 67](#_Toc185090249)

# **Введение**

В современных условиях автоматизация и цифровизация процессов становятся неотъемлемой частью жизни, охватывая практически все сферы деятельности. Одной из таких сфер является обеспечение безопасности на дорогах и контроль соблюдения правил дорожного движения. С ростом числа транспортных средств и необходимостью их эффективного контроля внедрение инновационных технологий приобретает особую важность. Одной из таких технологий является распознавание государственных регистрационных номеров, что позволяет автоматизировать процессы фиксации нарушений, улучшить управление дорожным движением и повысить безопасность на дорогах.

Актуальность разработки веб-сервиса для распознавания регистрационных номеров обусловлена потребностью в автоматизации идентификации автомобилей в различных системах, включая системы безопасности, мониторинг дорожного движения и парковочные комплексы. Традиционные методы контроля часто оказываются недостаточно эффективными, требуют значительных затрат времени и человеческих ресурсов, а также подвержены человеческому фактору. Автоматическое распознавание номерных знаков, напротив, позволяет повысить точность и скорость обработки данных, минимизируя ошибки, что повышает надежность и безопасность.

Целью данной преддипломной практики является разработка веб-сервиса, который будет эффективно распознавать регистрационные номера транспортных средств на изображениях и видеопотоках. Сервис должен иметь удобный интерфейс, поддержку хранения и управления результатами распознавания, а также гибкую настройку для работы в различных условиях, включая низкое качество изображений и плохое освещение.

В процессе преддипломной практики изучаются технологии компьютерного зрения и алгоритмы распознавания символов, которые используются для анализа изображений номерных знаков. Определяются подходящие программные инструменты, обеспечивающие точность и стабильность работы системы.

Результатом работы станет полностью функциональный веб-сервис, который обеспечит распознавание регистрационных номеров транспортных средств в реальном времени, а также удобные инструменты для управления данными. Это решение может быть применено в различных сферах, включая системы безопасности и автоматические парковки, что повышает эффективность работы с данными.

Таким образом, преддипломная практика направлена на создание программного решения, которое улучшает процесс распознавания номерных знаков и обеспечивает высокий уровень автоматизации и надежности в области контроля за транспортными средствами.

## **Анализ предметной области**

## **Описание предметной области**

Предметная область, связанная с веб-сервисом для распознавания государственных регистрационных номеров транспортных средств, охватывает широкий спектр задач, связанных с автоматической идентификацией и обработкой информации о транспортных средствах. Такие системы активно используются в современных городах, на автомагистралях, в системах контроля доступа, управления дорожным движением, на парковках, а также в частных и государственных охранных системах. Они играют ключевую роль в повышении эффективности транспортных систем, улучшении безопасности дорожного движения и автоматизации различных процессов, связанных с транспортными средствами.

Распознавание номерных знаков является неотъемлемой частью систем контроля дорожного движения, обеспечения правопорядка, мониторинга транспортных потоков и автоматизации логистических операций. Например, в городских системах видеонаблюдения данная технология позволяет в режиме реального времени определять автомобили, нарушающие правила дорожного движения, такие как превышение скорости, проезд на красный свет или нахождение в запрещенной зоне. Эти данные могут быть немедленно переданы в соответствующие органы для принятия оперативных мер. В дополнение к этому, системы распознавания могут применяться для автоматизации платных парковок, где вход и выход транспортных средств осуществляется на основе распознавания их номерных знаков, что позволяет минимизировать затраты на обслуживание и повысить удобство для пользователей.

Основной технологической основой этой предметной области является компьютерное зрение – технология, которая позволяет компьютерам «видеть» и интерпретировать визуальные данные, как это делает человеческий глаз. В случае с распознаванием номерных знаков, эта технология направлена на извлечение текстовой информации с изображений автомобилей и анализ видеопотоков, поступающих от камер наблюдения или специализированных устройств. Веб-сервисы, основанные на таких технологиях, должны обладать высокой степенью точности и скорости работы, чтобы эффективно решать задачи в реальном времени.

Система распознавания номерных знаков имеет широкий спектр применения. Одним из ключевых сценариев использования является обеспечение контроля доступа на охраняемые территории, такие как частные объекты, склады, бизнес-центры или жилые комплексы. В этом случае веб-сервис обрабатывает изображение автомобиля, распознает номерной знак, проверяет его в базе данных разрешенных номеров, и, при совпадении, автоматически открывает шлагбаум или ворота. Это значительно упрощает и ускоряет процесс въезда, устраняя необходимость в ручной проверке документов или пропусков.

Еще одним важным направлением использования данной технологии является обеспечение безопасности на дорогах. Системы распознавания номерных знаков активно применяются в системах фиксации нарушений правил дорожного движения. Например, камеры, установленные вдоль дороги или на перекрестках, могут фиксировать превышение скорости, проезд на запрещающий сигнал светофора или нарушение других правил. Веб-сервис в этом случае не только распознает номерной знак автомобиля-нарушителя, но и передает информацию в соответствующие государственные органы для принятия мер. Такие системы значительно повышают эффективность работы дорожной полиции, позволяют быстро реагировать на нарушения и обеспечивают высокий уровень контроля на дорогах.

Технологии, используемые для распознавания номерных знаков, строятся на сочетании методов машинного обучения, глубокого обучения и компьютерного зрения. Основная задача состоит в том, чтобы система могла распознавать символы на номерных знаках транспортных средств с высокой степенью точности, вне зависимости от условий съемки, таких как плохое освещение, непогода или нестандартные углы обзора. Для этого используются нейронные сети, которые обучаются на огромных наборах данных, содержащих изображения номерных знаков в различных условиях. В процессе обучения система совершенствует свои алгоритмы, что позволяет ей надежно распознавать знаки даже в самых сложных условиях.

Одной из особенностей таких систем является работа в реальном времени. Это означает, что веб-сервис должен обрабатывать поступающие данные практически мгновенно, чтобы результат был готов к использованию уже через доли секунд после захвата изображения. Это критически важно в сценариях с интенсивным движением, таких как контроль трафика на загруженных дорогах или автострадах, где промедление может привести к потере важных данных или снижению эффективности системы.

Для обеспечения работы веб-сервиса с высокой нагрузкой требуется использование производительных серверов и эффективное распределение ресурсов. Веб-сервисы для распознавания номерных знаков могут быть частью распределенных систем, где обработка данных осуществляется на нескольких серверах одновременно, что повышает масштабируемость и устойчивость к сбоям. Такая архитектура особенно важна в системах, работающих с большими объемами данных, например, в городских системах видеонаблюдения, где ежедневно обрабатываются миллионы изображений.

Одним из ключевых аспектов, который необходимо учитывать при проектировании и разработке подобных систем, является соблюдение норм и стандартов в области защиты персональных данных. Веб-сервисы, работающие с распознаванием номерных знаков, должны обеспечивать конфиденциальность данных, особенно если они используются для идентификации и отслеживания транспортных средств в режиме реального времени. Важно, чтобы система включала в себя механизмы шифрования данных, а также надежную аутентификацию пользователей, имеющих доступ к результатам работы системы. Это особенно актуально для правительственных и частных систем, которые работают с конфиденциальной информацией.

Особое внимание уделяется вопросам масштабируемости и гибкости системы. Важно, чтобы веб-сервис мог обрабатывать одновременно большое количество входящих запросов, что позволяет применять его в масштабных системах видеонаблюдения на дорогах или в крупных парковочных комплексах. Для этого система может быть развернута в облачной среде, что обеспечит возможность масштабирования в зависимости от нагрузки и позволит избежать сбоев при пиковых значениях входящих запросов.

В дополнение к вышеуказанным аспектам, современные системы распознавания номерных знаков должны поддерживать гибкую настройку под различные условия эксплуатации. Это может включать возможность настройки алгоритмов под разные типы номерных знаков, особенности их дизайна в разных странах, а также адаптацию под конкретные задачи. Например, в некоторых случаях требуется распознавать не только номера автомобилей, но и другие идентификационные признаки, такие как цвет машины, тип транспортного средства или наличие специальных знаков на кузове.

Важно подчеркнуть, что для обеспечения высокой точности работы системы необходима интеграция с большими объемами данных, на основе которых обучаются модели распознавания. Это требует применения методов глубокого обучения и использования тренировочных наборов данных, содержащих изображения номерных знаков с разными характеристиками: различными шрифтами, размерами, ориентацией и фоном. Кроме того, для повышения точности системы может быть использована технология адаптивного обучения, при которой алгоритм настраивается на конкретные особенности изображений, характерные для определенного региона или страны, например, специфический формат номерных знаков.

Таким образом, предметная область, связанная с разработкой веб-сервиса для распознавания государственных номерных знаков транспортных средств, представляет собой многослойную систему, включающую передовые методы машинного обучения и компьютерного зрения, а также требует тщательного учета требований по производительности, безопасности и конфиденциальности данных.

## **Обоснование актуальности разработки ПС**

В последние годы стремительное развитие технологий обработки изображений и компьютерного зрения привело к широкому внедрению систем автоматической идентификации и анализа объектов в различных сферах жизни. Одной из таких технологий является распознавание государственных регистрационных номеров транспортных средств. Эта задача особенно актуальна в условиях современных мегаполисов и растущего количества автомобилей на дорогах, что требует внедрения инновационных решений для обеспечения безопасности, эффективного управления трафиком и оптимизации процессов парковки и логистики.

Актуальность разработки веб-сервиса для распознавания государственных регистрационных номеров транспортных средств обусловлена несколькими важными факторами.

Во-первых, возросла необходимость автоматизации контроля транспортных потоков и повышения безопасности на дорогах. Системы видеонаблюдения и мониторинга активно используются для фиксации нарушений правил дорожного движения, таких как превышение скорости, проезд на красный свет, игнорирование знаков и разметки. Традиционные методы контроля требуют большого количества человеческих ресурсов и могут быть подвержены человеческому фактору, что снижает общую эффективность работы системы. Автоматизация процесса распознавания регистрационных номеров с помощью современных технологий позволяет исключить эти недостатки и значительно повысить производительность системы, ускоряя обработку данных и уменьшая вероятность ошибок.

Во-вторых, рост количества автомобилей и сложность управления транспортными потоками в крупных городах требует внедрения интеллектуальных систем, которые могут обеспечивать эффективное управление дорожным движением и автоматический контроль доступа. В этом контексте системы распознавания номерных знаков становятся важным инструментом для регулирования потока транспортных средств, например, на платных участках дорог, в зонах ограниченного доступа (въезд на территорию предприятий, жилых комплексов), на парковках и в логистических центрах. Веб-сервис, использующий технологию распознавания номерных знаков, позволяет эффективно контролировать въезд и выезд транспортных средств, а также собирать статистику о трафике, что помогает улучшить планирование и управление транспортной инфраструктурой.

В-третьих, актуальность разработки такой системы объясняется также экономической выгодой и возможностью сокращения издержек. В условиях, когда ручной труд становится дороже, компании и государственные учреждения стремятся к максимальной автоматизации своих процессов. Внедрение системы автоматического распознавания номеров позволяет не только сократить количество необходимого персонала, но и значительно уменьшить вероятность мошенничества и ошибок, связанных с человеческим фактором. Это, в свою очередь, снижает эксплуатационные расходы и повышает рентабельность таких систем.

Кроме того, веб-сервисы на базе технологий распознавания регистрационных номеров оказываются незаменимыми в системах безопасности. Они помогают контролировать движение транспортных средств на охраняемых объектах, фиксировать въезды и выезды в режиме реального времени, а также предотвращать несанкционированный доступ. В условиях увеличения угроз безопасности, таких как терроризм или кражи, автоматические системы распознавания номерных знаков становятся важным элементом системы контроля и мониторинга.

Еще одним важным аспектом актуальности разработки является широкая доступность технологий для создания таких систем. Библиотека OpenCV, являющаяся основой для обработки изображений в данной системе, активно развивается и предоставляет множество готовых инструментов для работы с компьютерным зрением. Благодаря этому разработчики могут создавать эффективные и высокопроизводительные решения, способные работать с большими объемами данных и обеспечивать высокую точность распознавания номерных знаков даже в условиях сложного освещения или плохого качества изображения. Это делает разработку подобных систем доступной и экономически целесообразной [7].

Также актуальность разработки веб-сервиса для распознавания номерных знаков обусловлена его высокой универсальностью и широким спектром применения в различных сферах. Он может быть интегрирован в парковочные системы для автоматического распознавания и проверки номерных знаков, что обеспечивает удобство и безопасность для водителей, а также способствует упрощению контроля за доступом на охраняемые территории. В системах безопасности такой сервис позволяет своевременно выявлять автомобили, находящиеся в розыске или подозреваемые в участии в противоправных действиях, что значительно ускоряет работу правоохранительных органов и повышает общий уровень безопасности в обществе.

Важным аспектом, подтверждающим актуальность разработки, является её потенциал в снижении затрат на человеческие ресурсы и минимизации рисков, связанных с человеческим фактором. Автоматизированное распознавание номерных знаков уменьшает вероятность ошибок, которые могут возникнуть при ручной обработке данных, а также обеспечивает более точное и бесперебойное выполнение поставленных задач. Это особенно актуально для крупных предприятий и организаций, где необходимо контролировать большой поток транспортных средств и обеспечивать высокую точность идентификации.

Не менее важным фактором является и интеграция с существующими информационными системами. Современные компании и государственные учреждения уже используют разнообразные системы видеонаблюдения и контроля транспортных потоков, и необходимость адаптации этих систем к новым стандартам и требованиям растет. Внедрение веб-сервиса для распознавания номерных знаков позволяет легко интегрировать его с уже существующими системами, что дает возможность оптимизировать текущие рабочие процессы без значительных затрат на реорганизацию инфраструктуры.

Кроме того, вопросы защиты и хранения данных также играют важную роль в актуальности разработки. С развитием цифровых технологий и увеличением объема обрабатываемой информации возрастают требования к безопасности и конфиденциальности данных. Разработка системы, соответствующей современным стандартам защиты информации, становится важной задачей, особенно в тех областях, где персональные данные, такие как регистрационные номера транспортных средств, требуют особого подхода к безопасности. Это делает создание надежной и защищенной системы распознавания номерных знаков актуальной задачей для многих организаций.

С учетом развития технологий искусственного интеллекта и машинного обучения, создание веб-сервиса для распознавания номерных знаков становится всё более актуальным и в контексте научно-технического прогресса. Современные алгоритмы глубокого обучения позволяют значительно повысить точность и надежность распознавания, адаптируя систему под различные условия эксплуатации. Это особенно важно для работы в сложных условиях освещения, при наличии препятствий на изображении, а также при анализе изображений с низким разрешением или шумом. Данный аспект подчеркивает необходимость и важность разработки системы, использующей передовые методы обработки изображений и машинного обучения для более точного и надежного распознавания номерных знаков в реальных условиях.

В современном мире развитие технологий распознавания также связано с глобальными вызовами в области безопасности и защиты данных. С ростом числа автомобилей и инфраструктуры повышается риск различных инцидентов, требующих быстрого и точного реагирования. В таких условиях автоматизированные системы распознавания, встроенные в веб-сервисы, позволяют более оперативно обрабатывать данные и принимать решения, что способствует снижению рисков и повышению общей безопасности на дорогах и охраняемых объектах.

Таким образом, разработка веб-сервиса для распознавания государственных регистрационных номеров транспортных средств является актуальной в свете современных тенденций автоматизации процессов, повышения требований к безопасности, необходимости оптимизации управления транспортом и снижения издержек. Данная технология предоставляет широкие возможности для использования в самых разных областях, от обеспечения безопасности на дорогах до автоматизации работы частных компаний и государственных учреждений.

## **Анализ методов, способов, подходов, методик, технологий**

Анализ методов, способов, подходов, методик и технологий, применяемых для распознавания государственных регистрационных номеров транспортных средств, является важным этапом в разработке веб-сервиса. Основная цель данного анализа – определить наиболее подходящие подходы и алгоритмы, обеспечивающие высокую точность, производительность и надежность системы. В основе решения данной задачи лежат технологии компьютерного зрения и искусственного интеллекта, а также методы предобработки и анализа изображений.

Прежде всего, ключевым методом для распознавания номерных знаков является технология компьютерного зрения, которая включает в себя набор алгоритмов и инструментов для автоматической обработки изображений. В рамках компьютерного зрения широко применяются методы сегментации изображений, поиска контуров и детектирования объектов. Сегментация позволяет выделить область изображения, содержащую номерной знак, что является важным первым шагом в процессе распознавания. Использование методов поиска контуров, таких как алгоритм Канни или метод Хафа, позволяет улучшить точность выделения нужных областей изображения, определяя границы и формы объектов.

Следующим важным подходом является применение алгоритмов машинного обучения для более точного распознавания символов на номерных знаках. В последние годы наибольшую популярность в этой области получили сверточные нейронные сети (CNN), которые показывают высокую эффективность в задачах классификации и детектирования объектов на изображениях. Сверточные нейронные сети обучаются на больших наборах данных, содержащих различные изображения номерных знаков, что позволяет им лучше адаптироваться к условиям реального мира. В процессе обучения нейронные сети способны распознавать как стандартные символы, так и те, которые имеют повреждения, перекрытия или нестандартные шрифты, что существенно повышает общую точность системы.

Еще одним важным подходом является использование методов оптического распознавания символов (OCR), которые позволяют извлекать текстовую информацию с изображений. Технология OCR является основным компонентом в системе распознавания номерных знаков, поскольку она непосредственно отвечает за преобразование изображений в текст. OCR-алгоритмы используют различные техники, включая сопоставление шаблонов, морфологический анализ и использование классификационных моделей для определения символов на изображении. В современных системах OCR также применяются методы машинного обучения для улучшения качества распознавания, что позволяет системе автоматически корректировать и дополнять результаты распознавания на основе накопленных данных.

Важной частью анализа является выбор методик предобработки изображений, так как качество исходных данных играет критическую роль в достижении высокой точности распознавания. В данном случае используются методы улучшения качества изображения, такие как коррекция контраста, фильтрация шумов и выравнивание гистограммы. Эти техники помогают улучшить видимость символов на номерных знаках и уменьшают влияние факторов, ухудшающих качество изображения, таких как плохое освещение, тени, отражения или другие визуальные шумы. Например, использование адаптивной бинаризации позволяет более точно выделить символы на изображении, что улучшает дальнейшую работу OCR-алгоритмов и моделей машинного обучения.

С точки зрения программных инструментов и библиотек, наиболее популярной технологией в области распознавания номерных знаков является OpenCV – открытая библиотека для компьютерного зрения, предоставляющая широкий спектр функций для обработки изображений и видео. OpenCV поддерживает интеграцию с языками программирования, такими как Python и C++, что делает её универсальной и легко интегрируемой в различные веб-сервисы и приложения. В OpenCV доступны инструменты для детектирования объектов, сегментации изображений, применения фильтров и других методов обработки, что делает её одной из самых подходящих технологий для создания веб-сервиса по распознаванию номерных знаков [10].

В контексте внедрения методов машинного обучения и глубокого обучения, популярными библиотеками являются TensorFlow и PyTorch, которые предоставляют широкие возможности для создания и обучения нейронных сетей. Эти библиотеки поддерживают работу с большими объемами данных и позволяют гибко настраивать модели, адаптируя их к конкретным задачам распознавания. Важно отметить, что использование глубоких сверточных сетей требует значительных вычислительных ресурсов, поэтому в архитектуре системы могут быть предусмотрены механизмы распределенных вычислений и оптимизации нагрузки.

Отдельное внимание в анализе уделяется подходам к масштабированию и оптимизации веб-сервиса. С учетом того, что система должна обрабатывать большие потоки данных в режиме реального времени, рекомендуется использовать технологии, обеспечивающие высокую производительность и устойчивость к нагрузкам. Это включает использование контейнеризации (например, с помощью Docker) для упрощения развертывания и масштабирования сервиса, а также применение технологий распределенного кэширования (например, Redis) для оптимизации хранения и обработки данных. Подходы к масштабируемости также включают использование микросервисной архитектуры, где каждый компонент системы может быть развернут и обновлен независимо от других, что улучшает гибкость и устойчивость всей системы.

Таким образом, анализ методов, подходов и технологий, применяемых для создания веб-сервиса распознавания государственных регистрационных номеров транспортных средств, показывает, что наиболее эффективными являются комбинации методов компьютерного зрения, машинного обучения и предобработки изображений.

## **Обзор существующих аналогов**

Обзор существующих аналогов веб-сервиса для распознавания государственных регистрационных номеров транспортных средств включает анализ различных программных решений, представленных на рынке. Изучение аналогов позволяет определить их основные преимущества и недостатки, что поможет при выборе функциональных компонентов, улучшении точности распознавания и обеспечении надежности разрабатываемой системы.

Одним из наиболее известных решений в данной области является OpenALPR – это коммерческое программное обеспечение для распознавания номерных знаков, которое активно используется в системах контроля доступа, парковках и системах безопасности (рисунок 1.1). OpenALPR поддерживает работу с различными камерами и может быть интегрирован с существующими системами видеонаблюдения. Программное обеспечение использует алгоритмы компьютерного зрения и OCR для распознавания номерных знаков с высокой точностью. Одним из ключевых преимуществ OpenALPR является его поддержка множества стран и форматов номерных знаков, что делает его универсальным инструментом. Однако, несмотря на свою эффективность, данное решение имеет недостаток – это его коммерческая модель. OpenALPR требует оплаты лицензии для использования, что может стать препятствием для мелких организаций или проектов с ограниченным бюджетом.

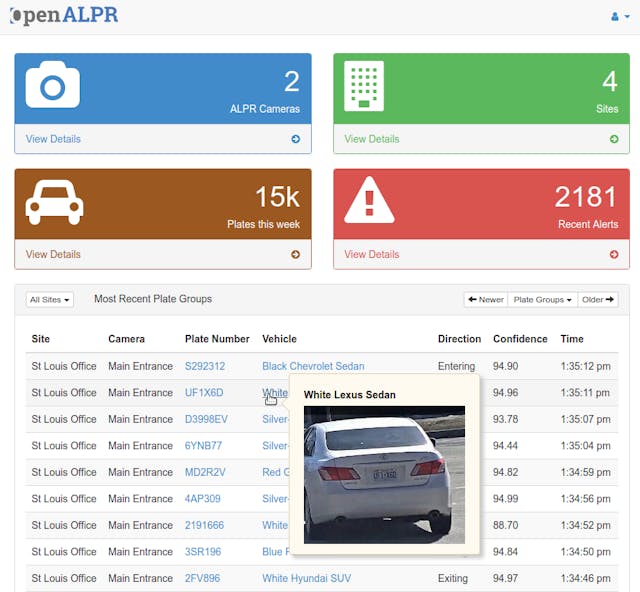


Рисунок 1.1 – интерфейс сервиса «OpenALPR»

Еще одним аналогом является ANPR Cloud – облачное решение для распознавания номерных знаков, которое предоставляет API для интеграции с различными системами (рисунок 1.2). ANPR Cloud может работать с изображениями и видеопотоками, предоставляя результаты в реальном времени. Одним из его преимуществ является возможность работы в облачной среде, что обеспечивает гибкость и масштабируемость. Однако, использование облачных решений может представлять собой вызов в странах с ограничениями на передачу данных за границу или строгими требованиями к защите данных. Также, для работы ANPR Cloud требуется стабильное интернет-соединение, что может быть недостатком в условиях работы в удаленных районах или при нестабильной связи.



Рисунок 1.2 – Сервис «ANPR CLOUD»

Еще одно популярное решение в данной области – Sighthound ALPR, которое предлагает как облачные, так и локальные версии программного обеспечения (рисунок 1.3). Оно также активно используется для распознавания номерных знаков в системах безопасности, мониторинга и контроля доступа. Программное обеспечение демонстрирует высокую точность распознавания и поддерживает работу с различными форматами номерных знаков. Одним из ключевых преимуществ Sighthound ALPR является его гибкость, так как оно может работать как в локальной сети, так и в облачной среде. Однако его основным недостатком является высокая стоимость лицензии, особенно для крупных установок, где требуется использование большого количества камер и серверов для обработки данных.



Рисунок 1.3 – Сервис распознавания номеров «Sighthound ALPR»

Анализ существующих аналогов показал, что большинство современных решений для распознавания номерных знаков имеют как свои сильные стороны, так и определенные недостатки. Важно отметить, что большинство коммерческих решений предлагают высокий уровень точности и широкий спектр функций, но их использование связано с финансовыми затратами, что может быть критически важным фактором для выбора в зависимости от бюджета проекта. Открытые и бесплатные решения менее затратны, но имеют свои ограничения, которые могут влиять на эффективность и надежность системы.

Таким образом, разработка собственного веб-сервиса для распознавания государственных регистрационных номеров транспортных средств должна учитывать недостатки существующих решений, такие как зависимость от стабильного интернет-соединения, высокие лицензионные расходы, а также ограничения по точности и скорости распознавания. Это позволит создать продукт, который будет конкурентоспособным и сможет удовлетворить потребности пользователей в различных сферах применения, включая контроль доступа, безопасность и мониторинг транспортных потоков.

## **Выбор и обоснование средств разработки**

Для разработки данного электронного средства выбраны технология ASP.NET core, язык программирования C#, паттерн проектирования MVC, среды разработки Microsoft Visual Studio Code и Microsoft Visual Studio 2019.

C# – объектно-ориентированный язык программирования. Разработан в 1998–2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft .NET Framework. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270 [1].

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML [5].

Переняв многое от своих предшественников – языков C++, Delphi, Модула, Smalltalk и, в особенности, Java – С#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, C# в отличие от C++ не поддерживает множественное наследование классов (между тем допускается множественное наследование интерфейсов).

ASP.NET Core – свободно-распространяемый кросс-платформенный фреймворк для создания веб-приложений с открытым исходным кодом. Данная платформа разрабатывается компанией Майкрософт совместно с сообществом и имеет большую производительность по сравнению с ASP.NET. Имеет модульную структуру и совместима с такими операционными системами как Windows, Linux и macOS.

Несмотря на то, что это новый фреймворк, построенный на новом веб-стеке, он обладает высокой степенью совместимости концепций с ASP.NET. Приложения ASP.NET Core поддерживают параллельное управление версиями, при котором разные приложения, работающие на одном компьютере, могут ориентироваться на разные версии ASP.NET Core. Это было невозможно в предыдущих версиях ASP.NET [4].

Microsoft Visual Studio – линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight [3].

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

Microsoft Visual Code – редактор исходного кода, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с открытым исходным кодом, но готовые сборки распространяются под проприетарной лицензией.

Многие возможности Visual Studio Code недоступны через графический интерфейс, зачастую они используются через палитру команд или JSON-файлы (например, пользовательские настройки). Палитра команд представляет собой подобие командной строки, которая вызывается сочетанием клавиш.

Model-View-Controller (MVC, «Модель-Представление-Контроллер») – схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер – таким образом, что модификация каждого компонента может осуществляться независимо.

## **Спецификация требований**

Спецификация требований к веб-сервису для распознавания государственных регистрационных номеров транспортных средств основывается на тщательном анализе существующих аналогов и их недостатков, а также учитывает цели и задачи разработки. Разрабатываемая система нацелена на создание высокоточной, производительной и масштабируемой платформы, способной автоматически идентифицировать номерные знаки на изображениях и видеопотоках в режиме реального времени. Основная концепция веб-сервиса заключается в предоставлении удобного и надежного решения, которое может интегрироваться с внешними системами и соблюдает требования безопасности и конфиденциальности данных.

Система должна обеспечивать обработку изображений и видеопотоков с извлечением регистрационных номеров и их преобразованием в текстовую информацию. Этот функционал является основой веб-сервиса и подразумевает использование технологий компьютерного зрения и алгоритмов OCR. Для повышения точности и надежности распознавания предполагается внедрение алгоритмов машинного и глубокого обучения, что позволит системе адаптироваться к различным условиям эксплуатации, включая плохое освещение, низкое качество изображения и нестандартные шрифты на номерных знаках. Важным аспектом является реализация API, который будет обеспечивать взаимодействие веб-сервиса с другими приложениями и системами. Такой API должен поддерживать операции по загрузке изображений, извлечению результатов распознавания, а также управлению параметрами обработки. Это позволит системе интегрироваться с внешними системами безопасности, парковочными решениями и системами контроля доступа.

Веб-сервис должен быть способен работать в условиях интенсивной нагрузки, обрабатывать несколько запросов одновременно и минимизировать задержки в обработке данных. Это означает, что он должен поддерживать многопоточность и параллельную обработку данных, что обеспечит высокую скорость и производительность, особенно при работе с видеопотоками в реальном времени. Такой подход предполагает использование распределенной архитектуры, что позволяет распределять вычислительные задачи между несколькими серверами или облачными ресурсами. Для достижения этого потребуется внедрение технологий контейнеризации и оркестрации, таких как Docker и Kubernetes, что обеспечит гибкость и возможность масштабирования системы в зависимости от объема данных и числа подключений [6].

Одной из важных задач является обеспечение хранения результатов распознавания в базе данных, которая должна включать информацию о номерных знаках, времени и дате распознавания, изображениях и других данных, которые могут быть полезны для дальнейшего анализа и статистики. Хранилище данных должно быть построено на основе реляционной или нереляционной базы данных, в зависимости от предпочтений и задач проекта. Например, реляционная база данных, такая как PostgreSQL или MySQL, будет удобна для структурированных данных, в то время как использование NoSQL решений, таких как MongoDB, может быть более подходящим для хранения изображений и видео [2].

Пользовательский интерфейс веб-сервиса должен обеспечивать интуитивно понятный доступ ко всем функциям системы, включая мониторинг работы, управление параметрами и просмотр результатов в реальном времени. Интерфейс должен быть адаптивным, чтобы обеспечивать удобство работы как на настольных компьютерах, так и на мобильных устройствах. Важно также предусмотреть многоуровневую систему доступа, которая позволит различным категориям пользователей, таким как операторы, администраторы и разработчики, получать доступ только к тем функциям, которые необходимы для их задач.

Входными данными для веб-сервиса являются изображения и видеопотоки, поступающие от камер видеонаблюдения или других устройств. Эти данные могут иметь различные форматы, включая статические изображения в JPEG и PNG, а также видеопотоки в MP4, AVI и других форматах. Для обеспечения гибкости работы система должна поддерживать преобразование и предобработку данных, таких как коррекция контраста, фильтрация шумов и адаптивная бинаризация. Это позволит улучшить качество изображений и повысить точность распознавания в сложных условиях, таких как плохое освещение или присутствие помех.

Выходными данными системы будут распознанные номерные знаки в текстовом формате, а также сопутствующая информация, такая как время и дата распознавания, качество обработки и вероятность точности. Веб-сервис должен обеспечивать хранение этих данных в структурированном виде, чтобы они могли быть легко доступны для дальнейшего анализа, фильтрации и отчётности. Данные также должны быть доступны для внешних систем через API, что позволит интегрировать их в более сложные системы аналитики и управления транспортными потоками.

С точки зрения временных характеристик, система должна обеспечивать обработку данных за минимальное время. Время распознавания одного изображения или кадра видеопотока не должно превышать 1-2 секунд, а задержка на обработку видеопотока должна быть минимальной, чтобы не превышать 2-3 секунд. Это критически важно для обеспечения работы в режиме реального времени, особенно в системах, требующих мгновенного реагирования, таких как контроль доступа или система видеофиксации нарушений.

Надежность веб-сервиса является ключевым требованием. Система должна обеспечивать доступность не менее 99%, что подразумевает минимальное количество простоев и перерывов в работе. Для обеспечения устойчивости к сбоям необходимо предусмотреть механизмы резервного копирования данных, что позволит избежать их потери в случае сбоев в работе серверов или других критических ситуаций. Встроенные механизмы обработки ошибок должны обеспечивать возможность идентификации и исправления ошибок в процессе распознавания, а также вести логи для анализа и улучшения работы системы.

Операционная среда для развертывания веб-сервиса должна включать серверы с многопоточными процессорами и не менее 16 ГБ оперативной памяти для обеспечения необходимой производительности. Поддержка операционных систем Linux и Windows позволит системе быть более гибкой и адаптируемой к различным средам эксплуатации. Важно также обеспечить совместимость с облачными платформами, такими как AWS, Azure или Google Cloud, что позволит развертывать систему в облаке и обеспечивать её масштабируемость в зависимости от объема данных и числа подключений.

Совместимость веб-сервиса с различными системами и базами данных является важным требованием, так как система должна поддерживать интеграцию с системами видеонаблюдения и базами данных, такими как PostgreSQL, MySQL или SQL Server. Веб-сервис должен быть кроссбраузерным, чтобы обеспечивать корректную работу в популярных браузерах, таких как Chrome, Firefox и Edge, что позволит пользователям работать с системой независимо от используемого браузера [8].

Безопасность системы является критически важным аспектом разработки. Все данные должны передаваться с использованием шифрования, например, через HTTPS, чтобы обеспечить их защиту от перехвата и несанкционированного доступа. Веб-сервис должен иметь встроенные механизмы аутентификации и авторизации, чтобы ограничить доступ только для авторизованных пользователей. Важно также предусмотреть меры по защите от внешних угроз, таких как DDoS-атаки и попытки взлома, чтобы обеспечить высокую степень безопасности и устойчивости системы.

Данная спецификация требований формирует комплексное представление о будущем веб-сервисе и определяет ключевые аспекты его функциональности, производительности, надежности и безопасности. Это обеспечит создание системы, которая будет соответствовать современным стандартам и требованиям, успешно решая задачи распознавания номерных знаков в различных сферах применения, включая безопасность, контроль доступа и управление транспортными потоками.

## **Моделирование предметной области**

## **Разработка функциональных моделей предметной области**

Функциональные модели предметной области обеспечивают наглядное представление ключевых процессов, связанных с работой веб-сервиса для распознавания государственных регистрационных номеров транспортных средств, а также взаимосвязи между его компонентами. Основной целью функционального моделирования является не только описание логики и последовательности работы системы, но и выявление всех необходимых этапов, участвующих в процессе обработки данных. Это помогает разработчикам и заинтересованным сторонам лучше понять, как будет функционировать система, и позволяет более точно определять требования к разработке и интеграции компонентов.

Разработка функциональных моделей начинается с определения базовых функций веб-сервиса. Главной функцией системы является автоматическое распознавание регистрационных номерных знаков, поступающих на изображениях и видеопотоках. Процесс распознавания можно разделить на несколько последовательных этапов. Сначала происходит предобработка изображения, которая включает улучшение качества изображения, фильтрацию шумов и корректировку контраста. Далее, система переходит к выделению области номерного знака на изображении, используя методы компьютерного зрения, такие как поиск контуров или сегментация изображения. Затем осуществляется сегментация символов, то есть разделение изображения на отдельные символы, которые будут подвержены дальнейшему распознаванию. На завершающем этапе применяется алгоритм OCR или модель глубокого обучения для идентификации символов и их преобразования в текст.

Функциональная модель верхнего уровня, которая описывает архитектуру и основные компоненты системы, должна четко определять логику взаимодействия этих компонентов. Веб-сервис состоит из нескольких ключевых модулей: модуль захвата изображений, модуль обработки изображений, модуль распознавания символов и модуль хранения данных. Модуль захвата изображений обеспечивает прием данных от внешних устройств, таких как камеры видеонаблюдения или загруженные пользователем изображения. После этого модуль обработки изображений выполняет предварительную обработку данных, применяя различные фильтры и алгоритмы для улучшения качества изображения, что помогает на последующих этапах распознавания. Модуль распознавания символов отвечает за распознавание символов на номерных знаках с использованием методов OCR и машинного обучения. Результаты передаются в модуль хранения данных, который осуществляет запись и управление данными в базе, чтобы они были доступны для анализа, отчетности и других функций.

Функциональная модель также описывает процессы взаимодействия пользователей и внешних систем с веб-сервисом. Пользователи могут взаимодействовать с системой через веб-интерфейс или API. Например, пользователь загружает изображение через веб-интерфейс, после чего оно поступает в обработку, а результаты возвращаются в виде текстовой информации. Аналогичным образом, внешние системы могут передавать изображения для обработки, а затем получать результаты в формате, удобном для их дальнейшего анализа или интеграции с другими системами, такими как базы данных для проверки регистрационных номеров или аналитические платформы для мониторинга транспортных потоков.

Важной частью функционального моделирования является описание процессов взаимодействия системы с пользователями и внешними системами. Например, пользователь может загрузить изображение через веб-интерфейс или API, после чего изображение поступает в обработку, и результаты распознавания возвращаются пользователю в виде текстовой информации. Взаимодействие с внешними системами может включать интеграцию с базами данных для проверки номерных знаков, получение данных от облачных сервисов и передачу результатов в другие аналитические системы.

Для более глубокого понимания функциональных процессов используется модель декомпозиции процессов, которая позволяет разбить каждый этап на подзадачи. Например, предобработка изображений может быть детализирована до уровня отдельных процедур, таких как адаптивная бинаризация, коррекция яркости, выравнивание гистограммы и шумоподавление. Каждый из этих подэтапов имеет свои входные и выходные данные, а также четкие связи с другими подзадачами. Это помогает определить требования к конкретным алгоритмам и компонентам системы, а также установить потенциальные точки интеграции и взаимодействия между разными модулями.

Таким образом, разработка функциональных моделей предметной области позволяет не только создать четкое представление о последовательности и логике работы веб-сервиса, но и детализировать процессы взаимодействия его компонентов. Это является основой для дальнейшего проектирования архитектуры системы, разработки базы данных и создания функциональных спецификаций требований, что будет детально рассмотрено в следующих подразделах. Функциональные модели способствуют выявлению возможных узких мест, рисков и потенциала для оптимизации работы системы, обеспечивая более эффективное планирование и реализацию проекта в целом.

## **Разработка и проектирование базы данных**

Разработка базы данных является важнейшей частью архитектуры веб-сервиса для распознавания государственных регистрационных номеров транспортных средств, так как именно в базе данных будут храниться результаты распознавания, изображения, информация о пользователях, а также другие критически важные данные. Процесс проектирования базы данных включает несколько ключевых этапов: определение структуры данных, выбор соответствующих типов данных, проектирование схемы базы данных, выбор системы управления базами данных (СУБД) и создание механизмов для обеспечения производительности и безопасности.

Первым шагом является определение основных данных, которые необходимо хранить в системе. Это включает информацию о транспортных средствах (распознанные регистрационные номера, дата и время фиксации, изображение транспортного средства), данные о пользователях и операторах системы, а также информацию, связанную с конфигурацией и логами системы (ошибки распознавания, метаданные изображений, качество обработки и другие параметры). Структура базы данных должна поддерживать возможность эффективного поиска, фильтрации и сортировки этих данных, а также их связность для облегчения анализа и отчетности.

Проектирование схемы базы данных основывается на принципах нормализации данных. База данных должна быть спроектирована так, чтобы минимизировать избыточность и исключить дублирование данных. Для этого создаются несколько таблиц, каждая из которых отвечает за хранение определенных данных. Например, можно выделить отдельные таблицы для хранения информации о распознанных номерных знаках, таблицы с изображениями, таблицы с информацией о пользователях, а также таблицы для журналирования действий системы. Эти таблицы будут связаны между собой через первичные и внешние ключи, что обеспечит целостность данных и их быструю доступность при выполнении запросов.

Особое внимание при проектировании базы данных уделяется выбору типов данных. Например, для хранения изображений могут использоваться такие типы данных, как BLOB (Binary Large Object), который позволяет сохранять изображения непосредственно в базе данных. Однако, с целью повышения производительности и снижения нагрузки на базу данных, часто рекомендуется хранить только пути к изображениям, которые сохраняются в файловой системе или облачном хранилище, а сами файлы не загружать в базу данных. Это позволяет значительно ускорить выполнение операций с базой данных и снижает её объем.

Выбор системы управления базой данных является важным этапом разработки. Для веб-сервиса по распознаванию номерных знаков транспортных средств важно использовать производительную и масштабируемую СУБД. Наиболее подходящими решениями могут быть реляционные базы данных, такие как PostgreSQL или MySQL. Они предоставляют мощные инструменты для работы с большими объемами структурированных данных, обладают широкими возможностями для индексации, обеспечения транзакционной целостности данных и поддержки сложных запросов. При необходимости масштабирования проекта можно рассмотреть и облачные решения, такие как Amazon RDS или Google Cloud SQL, которые обеспечат гибкость и возможность увеличения производительности без необходимости изменения основной структуры базы данных.

Производительность базы данных зависит не только от структуры данных, но и от правильного индексирования. Индексы создаются для часто запрашиваемых полей, таких как регистрационные номера или дата и время фиксации. Это значительно ускоряет выполнение запросов к базе данных, особенно в условиях больших объемов данных. Кроме того, необходимо предусмотреть механизмы оптимизации запросов и кэширования данных. Например, кэширование результатов часто выполняемых запросов с использованием таких решений, как Redis или Memcached, позволяет уменьшить нагрузку на базу данных и сократить время отклика веб-сервиса.

Безопасность данных также является важным аспектом проектирования базы данных. Поскольку система обрабатывает чувствительные данные, такие как регистрационные номера автомобилей, необходимо обеспечить их защиту от несанкционированного доступа и потерь. Это включает в себя использование шифрования для передачи данных между клиентом и сервером, а также шифрование данных, хранимых в базе. Кроме того, система должна предусматривать разграничение прав доступа для пользователей, что позволит ограничить доступ к критически важным данным только для авторизованных пользователей.

Наконец, важно разработать механизмы резервного копирования и восстановления базы данных, чтобы избежать потери данных в случае сбоев системы. Регулярные резервные копии должны автоматически создаваться и храниться в безопасном месте, чтобы в случае необходимости можно было быстро восстановить базу данных без значительных потерь.

Таким образом, проектирование базы данных для веб-сервиса по распознаванию регистрационных номеров транспортных средств является ключевым этапом, который определяет не только архитектуру системы, но и её производительность, надежность и безопасность. Правильное проектирование структуры данных, выбор соответствующей СУБД, обеспечение безопасности и производительности системы – всё это критически важно для успешной реализации проекта.

## **Разработка функциональной спецификации требований**

Разработка функциональных спецификаций требований является завершающим этапом моделирования предметной области для веб-сервиса распознавания регистрационных номерных знаков транспортных средств. Этот процесс направлен на формулировку детализированных технических требований, которые будут использоваться разработчиками для создания программного обеспечения, а также для проверки его работоспособности и соответствия целям проекта. Функциональные спецификации охватывают все аспекты работы системы, включая функциональные и нефункциональные требования, определение входных и выходных данных, а также требования к интерфейсам и интеграции.

Основное внимание в разработке функциональных спецификаций уделяется описанию функций системы, которые должны быть реализованы в веб-сервисе. Первостепенной задачей является обеспечение точного и быстрого распознавания регистрационных номерных знаков на изображениях и видеопотоках. Для этого в спецификациях подробно описываются требования к алгоритмам распознавания, включая применение методов предобработки изображений, таких как фильтрация шумов, коррекция контраста и адаптивная бинаризация. Спецификация также определяет, какие методы машинного и глубокого обучения должны быть использованы для сегментации и распознавания символов на номерных знаках. Это позволяет обеспечить высокую точность и надежность системы при обработке изображений в различных условиях, включая плохое освещение, плохое качество или нестандартные шрифты на номерах.

Спецификации должны также описывать процесс работы с API, который будет обеспечивать взаимодействие веб-сервиса с внешними приложениями и системами. API должен поддерживать стандартные методы HTTP, такие как POST и GET, для загрузки изображений, получения результатов распознавания и управления настройками обработки. В спецификациях указываются параметры запросов и ответов, а также формат данных (например, JSON), в котором будут передаваться результаты. Это позволяет обеспечить совместимость и удобство интеграции системы с другими решениями, такими как системы безопасности, парковочные системы или аналитические платформы [9].

Важным аспектом функциональных спецификаций является описание требований к интерфейсу пользователя. Веб-интерфейс должен быть интуитивно понятным и предоставлять доступ ко всем основным функциям системы, таким как загрузка изображений, просмотр результатов распознавания и управление настройками системы. Интерфейс должен быть адаптивным, чтобы обеспечивать удобство работы на различных устройствах, включая настольные компьютеры, планшеты и мобильные устройства. В спецификациях необходимо учесть требования к интерфейсу, такие как расположение элементов управления, доступность информации и время отклика системы.

Функциональные спецификации также включают требования к входным и выходным данным. Входными данными являются изображения и видеопотоки, поступающие от камер видеонаблюдения или других устройств захвата изображений. В спецификациях должны быть указаны поддерживаемые форматы файлов, такие как JPEG, PNG для изображений, и MP4, AVI для видеопотоков. Выходные данные включают текстовую информацию о распознанных номерных знаках, а также сопутствующую информацию, такую как время и дата распознавания, качество обработки и вероятность точности. Это позволяет пользователям и внешним системам анализировать результаты работы веб-сервиса и использовать их для дальнейшей обработки или интеграции.

Нефункциональные требования в спецификациях охватывают аспекты производительности, надежности и безопасности системы. Производительность системы должна быть высокой, чтобы обеспечить обработку данных в реальном времени. В спецификациях указывается максимальное время распознавания для одного изображения или кадра, а также максимальная задержка при обработке видеопотоков. Надежность веб-сервиса подразумевает высокую доступность (не менее 99%), наличие механизмов резервного копирования данных и обработки ошибок, а также ведение логов для последующего анализа и оптимизации системы. Безопасность является критически важным требованием, и спецификации должны включать меры по защите данных, такие как шифрование передачи данных, аутентификация и авторизация пользователей, а также защита от внешних угроз, таких как DDoS-атаки и попытки несанкционированного доступа.

Спецификации должны также учитывать требования к интеграции системы с базами данных и внешними системами. База данных должна поддерживать хранение результатов распознавания, изображений, данных о пользователях и другой связанной информации. В спецификациях указываются требования к структуре базы данных, типам данных и индексации, чтобы обеспечить быструю и эффективную работу системы. Для интеграции с внешними системами спецификации описывают, какие данные могут быть переданы другим приложениям, каким образом они будут передаваться и какие протоколы будут использоваться для обеспечения безопасности и совместимости данных.

Таким образом, разработка функциональных спецификаций требований завершает процесс моделирования предметной области, создавая подробное руководство для дальнейшей разработки веб-сервиса. Спецификации служат основой для проектирования, программирования и тестирования системы, а также для проверки её соответствия изначальным целям и требованиям проекта. Они обеспечивают единое понимание функционала и задач среди всех участников процесса разработки и гарантируют успешное выполнение проекта.

## **Проектирование и разработка программного средства**

## **Проектирование и разработка диаграммы вариантов использования**

Диаграмма вариантов использования, представленная для веб-сервиса распознавания государственных регистрационных номеров транспортных средств, демонстрирует основные сценарии взаимодействия пользователей с системой (рисунок 3.1). На диаграмме изображены два актера, которые представляют ключевые роли в системе: оператор и администратор. Каждый из этих актеров имеет свой набор функций, что позволяет четко разграничить задачи и процессы, которые они выполняют.

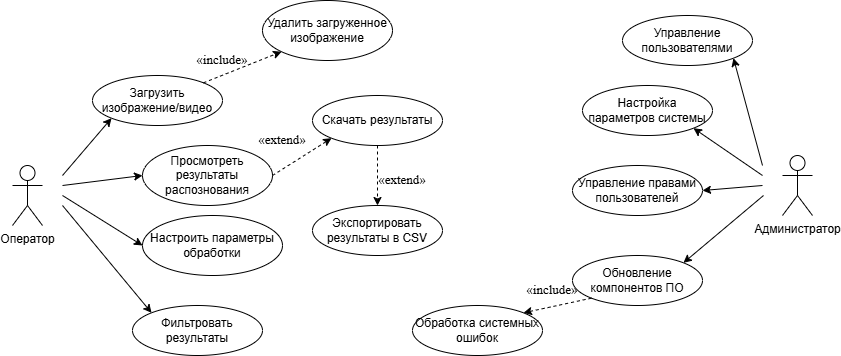


Рисунок 3.1 – диаграмма вариантов использования

Оператор взаимодействует с системой в рамках выполнения основных задач, связанных с обработкой и анализом изображений. Его работа начинается с загрузки изображения или видеопотока, что является начальной точкой для запуска процесса распознавания. Этот сценарий представлен в виде варианта использования «Загрузить изображение/видео». После успешной загрузки данных оператор может просматривать результаты распознавания. Этот вариант использования, обозначенный как «Просмотреть результаты распознавания», позволяет оператору видеть текстовые результаты, извлеченные из изображений, что обеспечивает базовую функциональность системы.

Кроме того, оператор может настроить параметры обработки, чтобы адаптировать работу алгоритмов под конкретные условия, такие как уровень освещения или качество изображения. Вариант использования «Настроить параметры обработки» описывает этот процесс. Важно отметить, что сценарий «Удалить загруженное изображение» тесно связан с настройкой параметров обработки, поэтому он обозначен на диаграмме через связь «include», указывающую, что данный шаг включен в общий процесс настройки системы. Оператор также имеет возможность фильтровать результаты по различным критериям, что позволяет ему упростить анализ данных и сосредоточиться на более релевантной информации.

Важной частью функционала является возможность скачивания результатов. Этот процесс представлен вариантом использования «Скачать результаты». В свою очередь, этот вариант использования связан со сценарием «Экспортировать результаты в CSV», который позволяет оператору сохранить результаты в удобном формате для дальнейшего анализа или интеграции с другими системами. Связь «extend» между этими вариантами использования указывает, что экспорт в CSV является дополнительной функцией, которая расширяет базовый сценарий скачивания результатов. Таким образом, оператор может не только просматривать и анализировать результаты в веб-интерфейсе, но и экспортировать их для работы в других аналитических инструментах.

Администратор, в свою очередь, выполняет функции управления и настройки системы. Он отвечает за управление пользователями, настройку параметров работы системы и установку прав доступа. Диаграмма показывает, что администратор может выполнять такие задачи, как «Управление пользователями», «Настройка параметров системы» и «Управление правами пользователей». Эти процессы обеспечивают надежность и безопасность системы, позволяя ограничить доступ к различным функциям в зависимости от роли пользователя.

Особое внимание уделяется обработке системных ошибок, что является одной из задач администратора. Вариант использования «Обработка системных ошибок» связан со сценарием «Обновление компонентов ПО», который указывает на необходимость исправления ошибок и обновления системы. Этот процесс обозначен на диаграмме с помощью связи «include», что подчеркивает тесную связь между обработкой ошибок и обновлением программного обеспечения.

Таким образом, представленная диаграмма вариантов использования помогает создать полное понимание функциональности веб-сервиса для распознавания номерных знаков. Она иллюстрирует не только основные сценарии взаимодействия, но и внутренние зависимости между ними, что позволяет проектировать систему, учитывая особенности работы каждого пользователя и их роль в обеспечении стабильной и надежной работы системы.

## **Проектирование и разработка диаграммы деятельности**

Диаграмма деятельности, представленная для веб-сервиса распознавания регистрационных номерных знаков транспортных средств, детально описывает последовательность операций, связанных с обработкой изображений, взаимодействием с базой данных и выводом результатов для оператора. Она визуализирует весь цикл работы системы, от начальной загрузки данных до финальной выдачи результатов, показывая, как различные процессы связаны между собой и какие действия предпринимаются в зависимости от успешности выполнения каждой операции. Диаграмма деятельности программы распознавания государственных номеров представлена на рисунке 3.2.

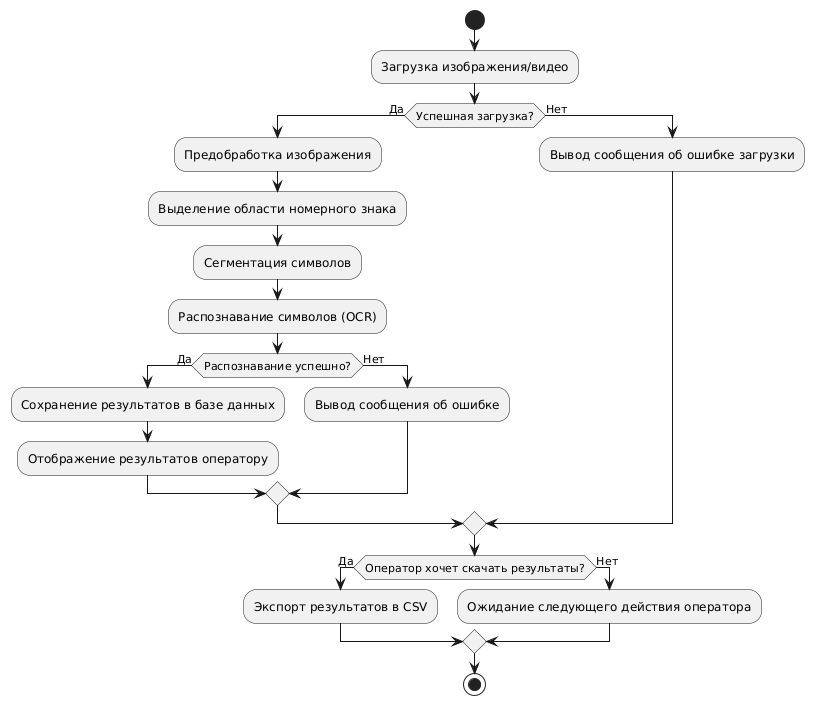


Рисунок 2.2 – Диаграмма деятельности

Процесс начинается с того, что оператор загружает изображение или видеопоток в систему. Этот этап играет решающую роль, поскольку он является точкой входа в работу веб-сервиса. Система сразу же проверяет успешность загрузки данных. Эта проверка необходима для обеспечения корректности данных, так как только корректно загруженные изображения могут быть обработаны на следующих этапах. Если загрузка изображения прошла успешно, процесс продолжается, если же произошла ошибка (например, файл был поврежден или формат изображения не поддерживается), система мгновенно уведомляет оператора об ошибке и не позволяет продолжить работу с некорректными данными.

В случае успешной загрузки начинается этап предобработки изображения. Он включает в себя серию операций по улучшению изображения для дальнейшей обработки: корректировку контраста, фильтрацию шумов, выравнивание яркости и резкости, а также другие необходимые манипуляции. Основной целью предобработки является улучшение видимости и четкости номерного знака, что критически важно для повышения точности последующего распознавания. Например, в условиях низкого освещения или плохого качества изображения предобработка может стать определяющим фактором успеха распознавания.

Следующий этап — выделение области номерного знака. Это процесс, в котором система определяет местоположение номерного знака на изображении, используя методы компьютерного зрения, такие как алгоритмы поиска контуров или модели глубокого обучения. На этом этапе важно обеспечить точность и надежность выделения, так как любые ошибки могут негативно сказаться на дальнейших шагах. Номерной знак изолируется от остальной части изображения, чтобы сосредоточить все вычислительные ресурсы на анализе именно этой области. Это не только повышает точность распознавания, но и улучшает общую производительность системы, минимизируя ненужные вычисления.

После выделения области номерного знака система переходит к этапу сегментации символов. Сегментация символов представляет собой процесс разделения изображения номерного знака на отдельные символы, такие как цифры и буквы. Это позволяет каждой части знака быть распознанной отдельно, что значительно повышает точность извлечения информации. Сегментация может использовать различные методы, включая алгоритмы анализа пиксельных связей или модели нейронных сетей, обученные для идентификации отдельных символов на изображении.

Основной этап работы системы — распознавание символов с использованием технологий оптического распознавания символов (OCR). На этом этапе система анализирует каждый символ, выделенный на этапе сегментации, и преобразует его в текстовый формат. OCR использует заранее обученные модели для точного определения символов, а также включает корректирующие механизмы для минимизации ошибок. Например, система может учитывать типичные формы и размеры символов на номерных знаках, что позволяет улучшить точность распознавания, особенно при обработке нестандартных или слегка поврежденных знаков.

После выполнения распознавания система проверяет успешность процесса. Если все символы распознаны корректно, результаты сохраняются в базе данных. База данных играет ключевую роль в системе, так как она используется для хранения всех данных, полученных в процессе работы: от исходных изображений до конечных результатов распознавания, а также метаданных, таких как дата и время обработки, качество изображения и вероятность точности распознавания. Система сохраняет данные в структурированном виде, что позволяет легко и быстро осуществлять доступ к ним для анализа или отчётности.

Если распознавание оказалось успешным, оператору отображаются результаты. Это может быть текстовое представление регистрационного номера, извлеченного из изображения, а также дополнительная информация о качестве распознавания и вероятности точности. Пользовательский интерфейс предоставляет возможность анализировать результаты в удобной форме, что позволяет оператору быстро оценивать успех работы системы и принимать решения о дальнейшем использовании данных.

Если на каком-либо этапе процесса распознавания произошла ошибка, система уведомляет об этом оператора, предоставляя информацию о типе и причине ошибки. Это может быть полезно для корректирующих действий, таких как повторная загрузка изображения или изменение параметров обработки. Аналогично, если загрузка изображения изначально была неуспешной, система сразу же уведомляет оператора об ошибке, что предотвращает дальнейшие попытки обработки некорректных данных.

После успешного отображения результатов оператор может выбрать дальнейшие действия. Если оператор решает скачать результаты распознавания, система предлагает экспорт данных в формате CSV, что удобно для последующего анализа в других системах или программных инструментах. Экспорт в CSV позволяет легко интегрировать результаты с другими аналитическими системами, а также создавать отчеты на основе полученных данных. Если оператор решает не скачивать результаты, система возвращается в состояние ожидания следующего действия оператора, что обеспечивает непрерывность и гибкость работы с системой.

Диаграмма деятельности помогает понять взаимодействие между пользователем и системой, а также отражает ключевые процессы, связанные с базой данных. Она демонстрирует, как данные проходят через разные этапы обработки, и подчеркивает важность правильной организации работы базы данных. База данных не только хранит результаты, но и играет центральную роль в обеспечении стабильной и надежной работы системы, предоставляя быстрый доступ к данным и поддержку последующих процессов анализа и отчетности.

## **Проектирование и разработка диаграммы классов**

Диаграмма классов, представленная для веб-сервиса распознавания регистрационных номерных знаков транспортных средств (рисунок 3.3), иллюстрирует основные сущности и их взаимосвязи в системе. Она представляет собой модель, которая описывает структуру данных и их взаимосвязь, а также определяет основные функции и операции, доступные для работы с этими данными.

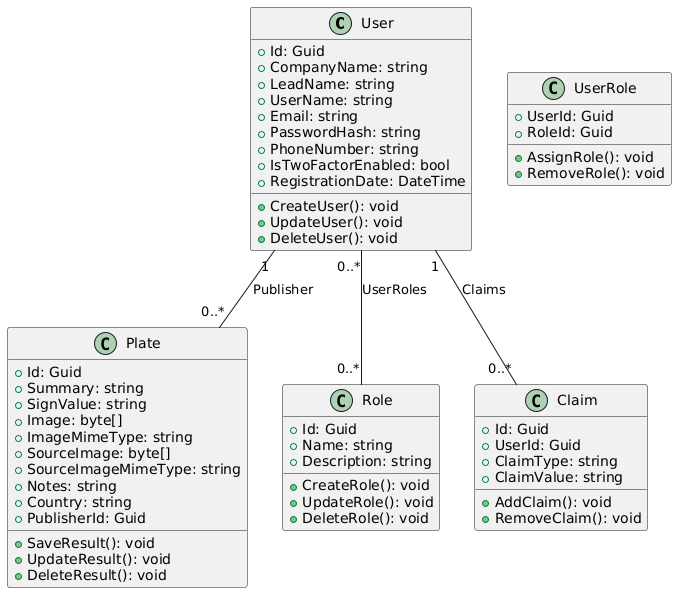


Рисунок 3.3 – Диаграмма классов

Центральной частью системы является класс User, который представляет учетные записи пользователей, работающих в системе. В этом классе хранится вся основная информация о пользователях, включая их уникальный идентификатор (Id), имя компании (CompanyName), имя руководителя (LeadName), имя пользователя (UserName), а также контактные данные, такие как электронная почта (Email) и номер телефона (PhoneNumber). Для обеспечения безопасности в классе предусмотрены такие атрибуты, как хеш пароля (PasswordHash), штамп безопасности (SecurityStamp) и параметр включения двухфакторной аутентификации (IsTwoFactorEnabled). Дата регистрации пользователя (RegistrationDate) позволяет отслеживать время создания учетной записи, что важно для анализа активности и учета пользователей. Методы класса User реализуют базовые операции управления учетными записями: создание, обновление и удаление пользователей. Это обеспечивает возможности администрирования и поддержания актуальности данных в системе.

Класс Plate представляет сущность, связанную с результатами распознавания регистрационных номеров транспортных средств. Он включает в себя несколько атрибутов, которые описывают результаты обработки изображений, такие как идентификатор записи (Id), краткое описание (Summary) и значение распознанного номерного знака (SignValue). Хранение изображений в этом классе реализовано через два атрибута: Image и SourceImage, которые представляют собой изображение номерного знака и исходное изображение соответственно. Дополнительно для каждого изображения сохраняется его MIME-тип (ImageMimeType и SourceImageMimeType), что позволяет определить формат файла и обеспечить корректное отображение данных. Другие атрибуты класса включают заметки (Notes) и страну (Country), что позволяет хранить дополнительную информацию о распознанных номерах. Важным элементом является атрибут PublisherId, который связывает результаты распознавания с конкретным пользователем, загрузившим изображение. Методы класса Plate реализуют функции сохранения, обновления и удаления данных о результатах, что позволяет управлять всеми этапами работы с распознанными номерными знаками.

В системе предусмотрен гибкий механизм управления ролями пользователей, который реализован через класс Role. Этот класс содержит такие атрибуты, как идентификатор роли (Id), имя роли (Name) и описание роли (Description), что позволяет четко идентифицировать и классифицировать права доступа для различных категорий пользователей. Для каждой роли могут быть установлены определенные права, которые влияют на доступ к функциям веб-сервиса. Например, администраторы могут иметь расширенные права доступа, такие как управление пользователями и настройками системы, в то время как операторы будут ограничены только доступом к функциям загрузки и распознавания изображений. Методы класса Role позволяют создавать, обновлять и удалять роли, что делает управление доступом гибким и настраиваемым в зависимости от бизнес-логики системы.

Связь между пользователями и ролями реализована через промежуточный класс UserRole, который позволяет каждому пользователю иметь несколько ролей, а каждой роли — быть назначенной нескольким пользователям. Класс UserRole содержит два атрибута: идентификатор пользователя (UserId) и идентификатор роли (RoleId). Это делает возможным хранение и управление множественными ролевыми ассоциациями для каждого пользователя. Методы этого класса обеспечивают назначение и удаление ролей у пользователей, что способствует гибкому управлению доступом в системе. Такая реализация позволяет создавать сложные схемы ролевого доступа и обеспечивает безопасность данных, так как доступ к определенным функциям и данным может быть ограничен в зависимости от роли пользователя.

Класс Claim представляет собой механизм управления утверждениями, которые являются дополнительными атрибутами, присваиваемыми пользователям для более точного контроля доступа и выполнения сложных правил авторизации. Утверждения (claims) позволяют системе добавлять дополнительные уровни безопасности, которые могут быть использованы для разграничения доступа к различным функциям. Атрибуты класса включают идентификатор утверждения (Id), идентификатор пользователя (UserId), тип утверждения (ClaimType) и его значение (ClaimValue). Методы этого класса позволяют добавлять и удалять утверждения для пользователей, что расширяет возможности настройки системы безопасности. Утверждения могут использоваться, например, для определения специальных прав доступа к конкретным данным или функциям, таких как доступ к конфиденциальной информации или выполнение административных операций.

Диаграмма классов демонстрирует не только ключевые сущности системы, но и их взаимосвязи, которые реализуются с использованием отношений "один ко многим". Например, один пользователь может быть связан с несколькими результатами распознавания в классе Plate, а также иметь несколько ролей и утверждений. Такая структура позволяет обеспечить целостность данных и четкую организацию работы системы, что упрощает реализацию бизнес-логики и поддерживает устойчивость архитектуры.

Проектирование диаграммы классов является важным шагом в создании архитектуры веб-сервиса, так как она формирует основу для последующей реализации функционала и взаимодействия с базой данных. Четкое понимание структуры данных и логики их использования помогает разработчикам более эффективно реализовать функции системы, обеспечивая при этом высокую производительность, надежность и безопасность приложения.

## **Проектирование и разработка алгоритмов работы программы**

В рамках разработки веб-сервиса распознавания регистрационных номерных знаков транспортных средств были спроектированы и описаны три ключевых алгоритма работы системы: алгоритм добавления нарушения, алгоритм авторизации и алгоритм поиска. Эти алгоритмы являются основой для функциональности приложения и позволяют обеспечивать обработку данных, управление доступом и поиск информации в системе.

Алгоритм добавления нового нарушения в систему представляет собой последовательность шагов, начиная с инициации процесса и заканчивая сохранением данных в базе (рисунок 3.4).

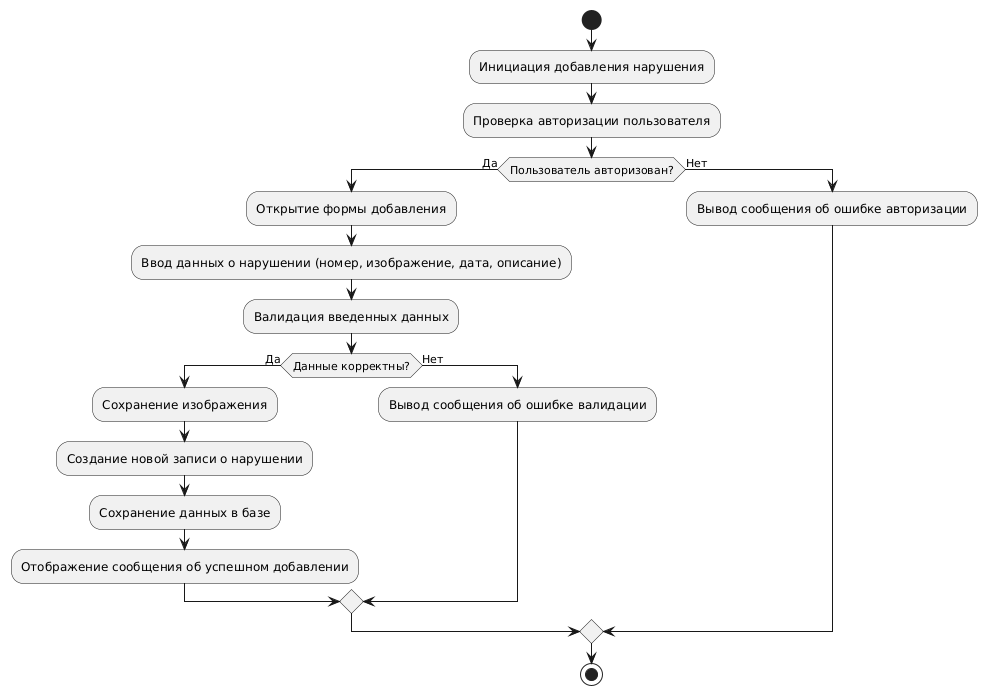


Рисунок 3.4 – Алгоритм добавления нарушения

Процесс начинается с инициации добавления нового нарушения. На первом этапе выполняется проверка авторизации пользователя, чтобы убедиться, что у него есть права на выполнение этой операции. Если пользователь авторизован, система открывает форму для ввода данных о нарушении, включая номер транспортного средства, изображение, дату и описание инцидента. Затем данные проходят этап валидации, в ходе которого проверяется их корректность. В случае успешной валидации происходит сохранение изображения и создание новой записи о нарушении в базе данных. По завершении этих операций система отображает сообщение об успешном добавлении. Если в ходе валидации обнаруживаются ошибки, пользователю выдается соответствующее сообщение. В случае неуспешной авторизации система также сразу выдает сообщение об ошибке и прерывает процесс добавления.

Алгоритм четко структурирует добавление нарушений, учитывая как проверку прав доступа, так и корректность введенных данных, что помогает обеспечить надежность и целостность данных в системе.

Алгоритм авторизации пользователя является важным компонентом безопасности системы, обеспечивающим проверку доступа и аутентификацию пользователей (рисунок 3.5).

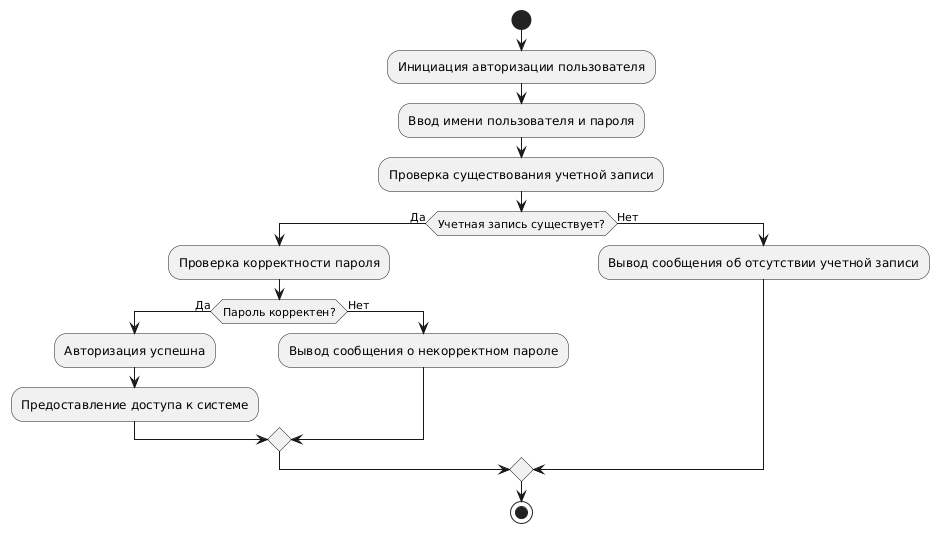


Рисунок 3.5 – алгоритм авторизации

Процесс начинается с инициации авторизации, когда пользователь вводит свои учетные данные, включая имя пользователя и пароль. Система выполняет проверку на наличие учетной записи, соответствующей введенным данным. Если учетная запись не найдена, пользователю выдается сообщение об ошибке, и процесс завершается. Если учетная запись существует, система проверяет корректность введенного пароля. В случае успешной проверки пользователь получает доступ к системе, и процесс завершается. Если же пароль некорректен, система выводит сообщение об ошибке и прерывает процесс авторизации.

Алгоритм построен таким образом, чтобы минимизировать риски несанкционированного доступа, обеспечивая надежную проверку учетных данных и безопасный вход в систему.

Алгоритм поиска в системе позволяет пользователю находить информацию на основе различных параметров, таких как регистрационный номер, дата нарушения или страна (рисунок 3.6).

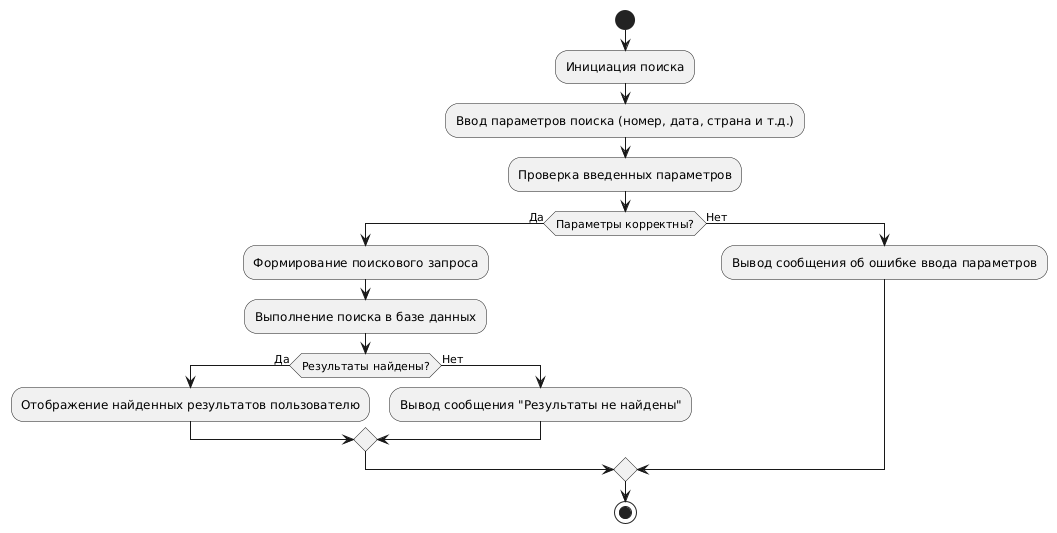


Рисунок 3.6 – Алгоритм поиска

Инициация процесса начинается с ввода параметров поиска пользователем. Затем система выполняет проверку корректности введенных параметров. Если параметры корректны, формируется поисковый запрос, который отправляется в базу данных. В случае успешного нахождения результатов система отображает их пользователю. Если же совпадений в базе данных не найдено, выводится сообщение "Результаты не найдены". Если на этапе проверки параметров вводятся некорректные данные, пользователю сразу выдается сообщение об ошибке, что позволяет ему внести изменения и повторить попытку поиска.

Алгоритм поиска спроектирован так, чтобы обеспечить быстрое и точное нахождение информации, что позволяет пользователю эффективно работать с данными в системе.

Представленные алгоритмы обеспечивают базовую функциональность веб-сервиса, необходимую для добавления, поиска и управления данными о нарушениях, а также для безопасной авторизации пользователей. Они обеспечивают не только удобство использования, но и высокую степень надежности и защиты данных, что критически важно для работы системы. Диаграммы деятельности четко отражают логику каждого алгоритма, показывая все ключевые этапы, условия и возможные исходы процессов.

## **Проектирование и разработка базы данных**

Проектирование и разработка базы данных для веб-сервиса распознавания регистрационных номерных знаков транспортных средств (рисунок 3.7) является ключевым этапом, поскольку она служит основным хранилищем данных, которые формируются в ходе обработки изображений. Основные сущности и их взаимосвязи представлены на диаграмме, что позволяет наглядно понять структуру базы данных и логику её работы.

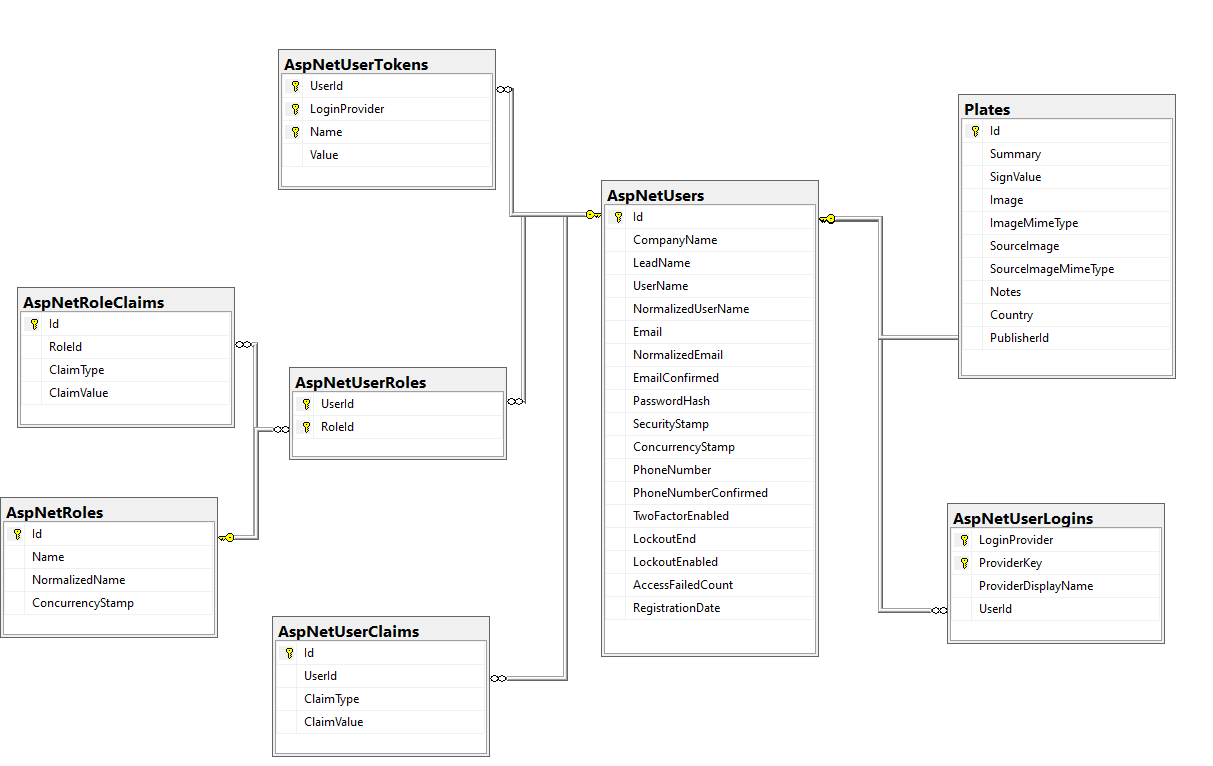


Рисунок 3.7 – база данных веб-приложения

База данных состоит из нескольких таблиц, которые обеспечивают управление пользователями, аутентификацию, а также хранение результатов распознавания. Основными таблицами являются AspNetUsers, Plates, AspNetRoles, AspNetUserRoles, AspNetUserClaims, AspNetRoleClaims, AspNetUserTokens и AspNetUserLogins.

Таблица AspNetUsers является центральной таблицей системы аутентификации и авторизации. Она хранит информацию о пользователях системы, включая такие поля, как идентификатор пользователя (Id), имя компании (CompanyName), имя руководителя (LeadName), имя пользователя (UserName), нормализованное имя пользователя (NormalizedUserName), электронная почта (Email), нормализованная электронная почта (NormalizedEmail), а также различные параметры безопасности, такие как хеш пароля (PasswordHash), штамп безопасности (SecurityStamp), подтверждение номера телефона (PhoneNumberConfirmed) и включение двухфакторной аутентификации (TwoFactorEnabled). Эта таблица также хранит информацию о дате регистрации пользователя (RegistrationDate), что позволяет отслеживать время создания учетной записи. Вся информация в таблице используется для аутентификации и авторизации пользователей, а также для управления их доступом к различным функциям системы.

Таблица Plates отвечает за хранение результатов распознавания регистрационных номеров. В ней содержатся такие поля, как идентификатор (Id), краткое описание (Summary), значение распознанного номерного знака (SignValue), изображение номерного знака (Image), тип MIME изображения (ImageMimeType), исходное изображение (SourceImage), тип MIME исходного изображения (SourceImageMimeType), заметки (Notes), страна (Country) и идентификатор пользователя, загрузившего изображение (PublisherId). Эта таблица связана с таблицей AspNetUsers через поле PublisherId, что позволяет связывать конкретные результаты распознавания с пользователями, которые загружали изображения. Важно отметить, что хранение исходных и обработанных изображений в одной таблице упрощает управление данными и позволяет отслеживать их состояние на каждом этапе обработки.

Для обеспечения ролевого доступа и управления правами в системе используется набор таблиц, связанных с ролями и правами пользователей. Таблица AspNetRoles содержит информацию о ролях пользователей, включая их идентификатор (Id), имя роли (Name), нормализованное имя роли (NormalizedName) и штамп конкуренции (ConcurrencyStamp). Эта таблица связана с таблицей AspNetUserRoles, которая отвечает за установление связи между пользователями и ролями. Таблица AspNetUserRoles содержит идентификаторы пользователей (UserId) и ролей (RoleId), что позволяет определить, какой пользователь имеет какую роль в системе.

Таблица AspNetUserClaims хранит утверждения, связанные с пользователями. Она содержит такие поля, как идентификатор утверждения (Id), идентификатор пользователя (UserId), тип утверждения (ClaimType) и значение утверждения (ClaimValue). Эта таблица позволяет управлять дополнительными параметрами безопасности и доступом пользователей. Аналогично, таблица AspNetRoleClaims определяет утверждения, связанные с ролями, включая идентификатор утверждения (Id), идентификатор роли (RoleId), тип утверждения (ClaimType) и значение утверждения (ClaimValue). Эти таблицы обеспечивают гибкое управление правами и доступом в зависимости от ролей и утверждений, что позволяет адаптировать систему к различным требованиям безопасности и управления доступом.

Таким образом, проектирование и разработка базы данных для веб-сервиса охватывают несколько ключевых аспектов, включая управление пользователями, хранение результатов распознавания и обеспечение ролевого доступа. Структура базы данных учитывает необходимость эффективного и безопасного хранения данных, а также их доступности для последующего анализа и отчетности. Связи между таблицами реализованы с использованием первичных и внешних ключей, что обеспечивает целостность данных и возможность быстрого доступа к информации.

# **4 ТЕСТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА**

## **4.1 Выбор и обоснование видов тестирования**

Тестирование программного обеспечения является важным этапом в разработке, поскольку оно обеспечивает выявление и устранение ошибок, улучшает надежность и производительность системы. Основная цель тестирования заключается в подтверждении соответствия программного обеспечения установленным требованиям, проверке функциональности и предоставлении пользователям продукта высокого качества. Благодаря различным видам тестирования можно всесторонне проверить систему, начиная с отдельных компонентов и заканчивая общей интеграцией всех модулей.

Существуют следующие виды тестирования:

1. Модульное тестирование. Модульные тесты направлены на проверку отдельных компонентов приложения, таких как функции, классы и модули. Это тестирование выполняется на низком уровне, непосредственно в коде, и позволяет выявить дефекты в изолированных частях программы. Например, тестировались функции обработки изображений и алгоритмы распознавания символов, чтобы убедиться в корректности их работы [11].
2. Интеграционное тестирование. Интеграционное тестирование проверяет взаимодействие между различными компонентами системы, такими как клиентская и серверная части, база данных и внешние API. Этот вид тестирования помогает убедиться, что компоненты работают корректно при их объединении. Например, была протестирована работа REST API для передачи данных между клиентом и сервером, а также взаимодействие с базой данных [11].
3. Функциональное тестирование. Функциональные тесты ориентированы на проверку выполнения бизнес-требований системы. Они оценивают результаты выполнения функций без анализа промежуточных процессов. В случае веб-приложения тестировались основные операции, такие как загрузка изображений, распознавание номерных знаков, сохранение результатов и работа с пользовательским интерфейсом. Иногда возникает путаница между понятиями интеграционных и функциональных тестов, так как и те и другие требуют взаимодействия нескольких компонентов друг с другом. Разница в том, что интеграционный тест нужен просто чтобы убедиться, что вы можете отправлять запросы к базе данных, тогда как функциональный тест будет ожидать получения из базы данных определенного значения в соответствии с требованиями продукта [11].
4. Сквозное тестирование. Сквозное тестирование имитирует действия пользователя в контексте всего приложения. Оно обеспечивает проверку корректности работы системы в реальных сценариях. Например, были проверены такие сценарии, как авторизация пользователя, загрузка изображений, выполнение распознавания и экспорт данных. Это тестирование позволяет убедиться, что все этапы работы системы взаимодействуют друг с другом без сбоев. Сквозные тесты очень полезны, но их выполнение обходится довольно дорого, к тому же, когда они автоматизированы, такие тесты тяжело обслуживать. Рекомендуется иметь в наличии несколько основных сквозных тестов и активнее полагаться на более низкие уровни тестирования (модульные и интеграционные тесты), чтобы получать возможность быстро выявлять критические изменения [11].
5. Приемочное тестирование. Приемочные тесты — это формальные тесты, которые проверяют, отвечает ли система требованиям бизнеса. При этом во время тестирования должно быть запущено само приложение, и основное внимание уделяется воспроизведению поведения пользователей. В ходе этого тестирования возможен даже замер производительности системы, и в случае несоответствия установленным требованиям внесенные изменения могут быть отклонены [11].
6. Тестирование производительности. Тесты производительности оценивают, насколько система устойчива к нагрузкам и отвечает требованиям по скорости обработки данных. Для приложения были протестированы сценарии с высоким количеством запросов и большим объёмом данных, чтобы проверить стабильность и эффективность работы системы при пиковых нагрузках [11].
7. Smoke-тестирование. Smoke-тесты обеспечивают базовую проверку основных функций системы. Они позволяют быстро убедиться, что приложение работает в минимальном объёме, и его основные возможности функционируют корректно. Например, проверялась возможность загрузки изображения и получения результата распознавания. Smoke-тесты полезно запускать сразу после создания новой сборки (для определения, можно ли запускать более ресурсоемкие тесты) или сразу после развертывания (чтобы убедиться, что приложение работает правильно в новой, только что развернутой среде) [11].

Применение данных видов тестирования обеспечивают высокое качество работы приложения, его устойчивость к нагрузкам и соответствие заявленным требованиям. Проведение тестирования на всех уровнях позволяет выявить и устранить потенциальные проблемы до внедрения системы в эксплуатацию.

## **4.2 Результаты тестирования**

Тест-план представляет собой документ, в котором описывается стратегия тестирования, его цели, объем, ресурсы и график выполнения. В тест-плане указываются виды тестирования, методики их проведения, а также критерии завершения тестирования. Это позволяет структурировать процесс проверки и обеспечить его прозрачность для всех участников проекта.

Тест-кейс – это детализированный сценарий, описывающий последовательность действий для проверки определенной функциональности приложения. Каждый тест-кейс включает информацию о входных данных, ожидаемом результате, критериях прохождения и фактическом результате. Использование тест-кейсов позволяет стандартизировать процесс тестирования и минимизировать вероятность пропуска ошибок [12].

Тестирование программы происходило во время разработки а также по её завершении. Далее представлены тест-кейсы.

Таблица 4.1 – Тест-кейс наличия компонентов на главной странице

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Наличие главной страницы приложения и компонентов на ней   1. Открыть главную страницу сайта . 2. Проверить наличие компонентов на странице. | 1. Открывается главная страница сайта. 2. Отображаются следующие компоненты:  * Кнопка с названием сайта, перемещающая на главную страницу; * Кнопка для открытия окна входа; * Окно поиска нарушений по номеру автомобиля; * Последние зарегистрированные нарушения. | Успех |

Таблица 4.2 ‒ Тест-кейс наличия страницы входа

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Наличие страницы входа   1. Открыть страницу продукта входа, нажав на соответствующую кнопку 2. Проверить наличие компонентов на странице. | 1. Открывается страница входа. 2. Отображаются следующие компоненты:  * Поле для ввода логина; * Поле для ввода пароля; * Кнопка «Зарегистрироваться»; * Кнопка «Войти»; * Кнопка возврата на главную страницу. | Успех |

Таблица 4.3 ‒ Тест-кейс наличия страницы регистрации

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Наличие страницы регистрации   1. Открыть страницу входа. 2. Нажать кнопку «Зарегистрироваться». 3. Проверить наличие компонентов на странице. | 1. Открывается страница регистрации. 2. Отображаются следующие компоненты:  * Поле для ввода email; * Поле для ввода пароля; * Поле для повторения пароля; * Кнопка «Зарегистрироваться»; * Кнопка возврата на главную страницу. | Успех |

Таблица 4.4 ‒ Тест-кейс регистрации на сайте

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Проверка возможности регистрации на сайте   1. Открыть страницу регистрации. 2. Ввести данные во все поля. 3. Нажать кнопку «Зарегистрироваться». | 1. Открывается страница регистрации. 2. Отображаются компоненты страницы регистрации. 3. Пользователь успешно регистрируется. | Успех |

Таблица 4.5 ‒ Тест-кейс входа зарегистрированного пользователя

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Авторизация зарегистрированного пользователя   1. Открыть страницу входа. 2. Ввести данные во все поля. 3. Нажать кнопку «Войти» | 1. Открывается страница авторизации. 2. Отображаются компоненты страницы авторизации. 3. Пользователь успешно авторизируется. | Успех |

Таблица 4.6 ‒ Тест-кейс наличия проверки при создании пароля

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Наличие проверки при создании пароля   1. Открыть страницу входа. 2. Нажать кнопку «Зарегистрироваться». 3. Заполнить все поля с паролем «123456» 4. Нажать кнопку «Зарегистрироваться» | 1. Открывается страница регистрации. 2. После нажатия кнопки «Зарегистрироваться» учетная запись не создаётся и появляется надпись с указанием сложности пароля. | Успех |

Таблица 4.7 ‒ Тест-кейс наличия страницы зарегистрированных нарушений

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Наличие страницы зарегистрированных нарушений   1. Открыть страницу «Зарегистрированные нарушения». 2. Проверить наличие компонентов на странице | 1. Открывается страница зарегистрированных нарушений. 2. Отображаются следующие компоненты:  * Кнопка «Главная» для перехода на главную страницу; * Кнопка «Добавление нарушения»; * Таблица со столбцами «Название нарушения», «Заметки», «Номер автомобиля», «Фото автомобиля», «Зарегистрированное нарушение». | Успех |

Таблица 4.8 ‒ Тест-кейс добавления нарушения

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Добавление нарушения   1. Авторизироваться. 2. Открыть страницу зарегистрированных нарушений. 3. Нажать кнопку «Добавить нарушение». 4. Загрузить фото нарушения и заполнить поля. 5. Нажать кнопку загрузить. | 1. Открывается форма добавления нарушения. 2. Все поля корректно заполняются. 3. После нажатия кнопки «Загрузить» нарушение появляется в соответствующей таблице | Успех |

Таблица 4.9 ‒ Тест-кейс распознавания номера

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Распознавание номера   1. Открыть страницу зарегистрированных нарушений. 2. Нажать кнопку «Добавить нарушение». 3. Загрузить фотографию с автомобильным номером. 4. Нажать кнопку «Загрузить». | 1. Открывается страница зарегистрированных нарушений. 2. Отображается форма добавления нарушений. 3. После нажатия кнопки «добавить» нарушение появится в таблице с распознанным номером. | Успех |

Таблица 4.10 ‒ Тест-кейс редактирования нарушения

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Проверка возможности редактирования нарушения   1. Открыть страницу зарегистрированных нарушений. 2. Нажать иконку редактирования. 3. Изменить введенные значения. 4. Сохранить изменения. | 1. Открывается страница зарегистрированных нарушений. 2. При нажатии кнопки редактирования появляется возможность изменять параметры нарушения. 3. После сохранения изменений результат в таблице оставляет изменения. | Успех |

Таблица 4.11 ‒ Тест-кейс удаления нарушения

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Удаление нарушения   1. Открыть страницу зарегистрированных нарушений. 2. Нажать на иконку удаления нарушения. | 1. Открывается страница зарегистрированных нарушений. 2. После нажатия кнопки удаления данное нарушение исчезает из таблицы. | Успех |

Таблица 4.12 ‒ Тест-кейс поиска по номеру автомобиля

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Результат |
| Поиск по номеру автомобиля   1. Открыть главную страницу. 2. Ввести данные во все поля поиска. 3. Нажать кнопку «Поиск». | 1. Открывается главная страница. 2. После нажатия кнопки поиска выведутся все нарушения по введенному номеру автомобиля. | Успех |

Результаты тестирования показали, что приложение выполняет все основные функции в соответствии с заданными требованиями. Выявленные дефекты, такие как некорректная обработка некоторых форматов изображений и незначительные задержки при высоких нагрузках, были устранены. Повторное тестирование подтвердило исправление ошибок и стабильную работу системы.

## **4.3 Вывод тестирования**

Проведенное тестирование программного средства подтвердило его соответствие установленным требованиям и продемонстрировало готовность к внедрению в эксплуатацию. Комплексный подход к проверке функциональности, производительности и безопасности позволил оценить качество веб-приложения и выявить ключевые аспекты его работы, а также определить возможности для дальнейшей оптимизации.

Проведенные тесты подтвердили, что приложение соответствует заявленным требованиям и корректно выполняет основные задачи, включая обработку данных, управление пользователями и работу с нарушениями. Все функциональные компоненты, такие как страницы входа, регистрации, главная страница, а также механизмы добавления, редактирования, удаления и поиска нарушений, прошли проверку согласно разработанным тест-кейсам.

Результаты тестирования показали, что веб-приложение успешно справляется с обработкой входных данных. Высокая точность распознавания регистрационных номерных знаков была достигнута благодаря использованию алгоритмов компьютерного зрения, реализованных с помощью библиотеки OpenCV. Тестирование подтвердило корректность работы всех ключевых функций, таких как загрузка изображений, предобработка данных, распознавание номерных знаков, а также сохранение и управление результатами через пользовательский интерфейс. Отдельно было отмечено удобство использования интерфейса, который позволяет пользователям быстро и эффективно взаимодействовать с системой.

На основании результатов тестирования можно заключить, что разработанное программное средство соответствует заданным функциональным и нефункциональным требованиям. Проведенные проверки показали, что приложение готово к внедрению в эксплуатацию и способно обеспечить высокую производительность, надежность и удобство использования. Выявленные в ходе тестирования недостатки были успешно исправлены, а повторные проверки подтвердили их устранение.

# **РУКОВОДСТВО ПО ЭКСПЛУАТАЦИИ ПРОГРАММНОГО СРЕДСТВА**

## **Назначение и условия применения**

Разрабатываемое веб-приложение предназначено для регистрирования автомобилей с нарушениями и распознавания их номера. Основная цель данного программного средства повышение эффективности фиксирования нарушений автомобилей за счет возможности фиксирования при помощи пользователей а также автоматического распознавания номеров автомобилей по фото и последующим редактированием полученной информации.

Условия эксплуатации предполагают использования удобного серверного и клиентского оборудования, поддержку взаимодействия компонентов системы и интеграцию с базой данных.

Данное веб-приложение можно запустить через браузер на персональном компьютере.

## **Руководство пользователя**

При входе на сайт открывается главная страница, где есть иконка для авторизации, поиск по номеру автомобиля и список последних зарегистрированных нарушений (рисунок 5.1).

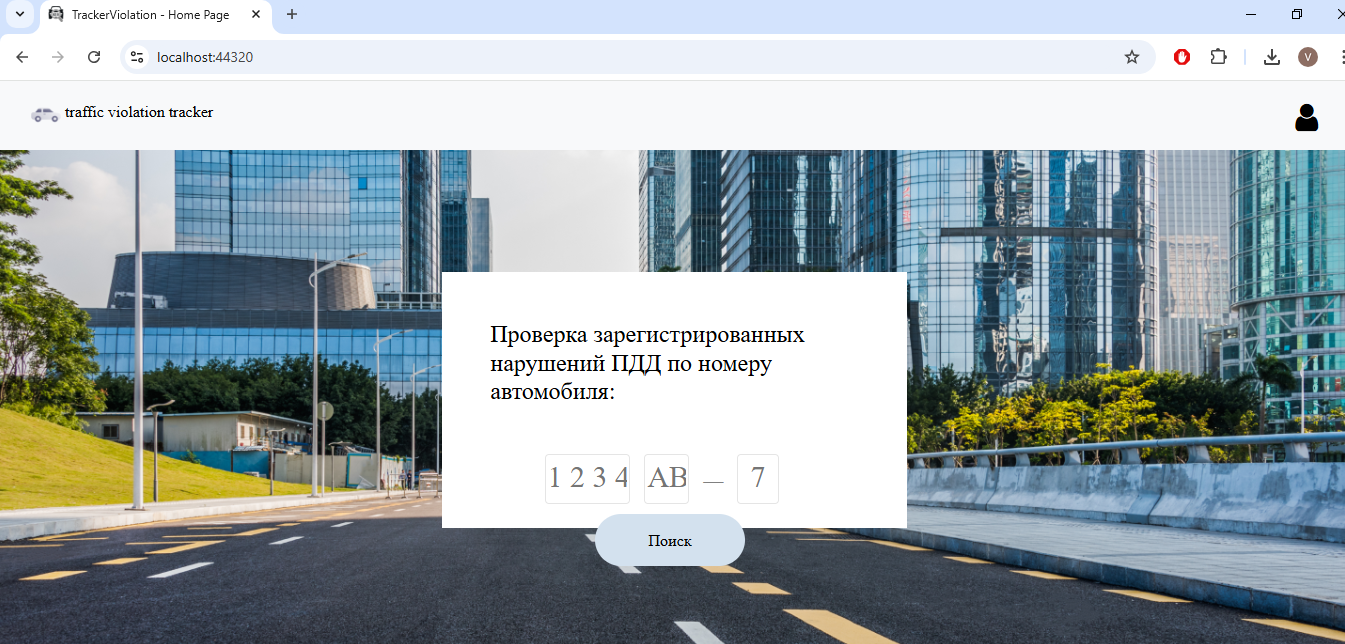


Рисунок 5.1 – Главная страница сайта

При нажатии кнопки авторизации открывается соответствующая страница (рисунок 5.2) где можно авторизироваться после заполнения необходимых полей.

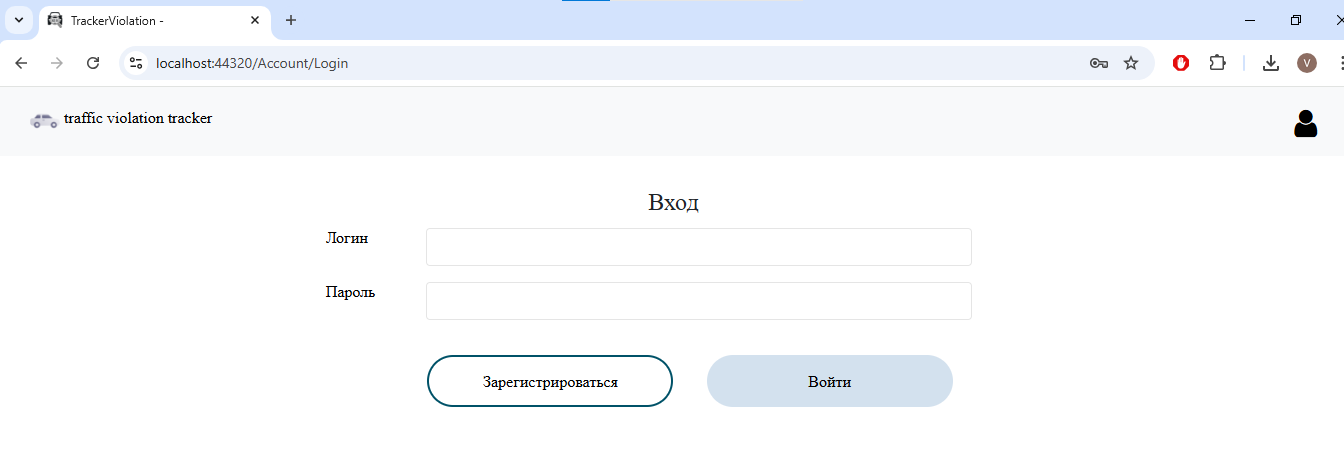


Рисунок 5.2 – Окно авторизации

Если пользователь еще не зарегистрирован можно нажать на кнопку «Зарегистрироваться» и перейти на соответствующее окно (рисунок 5.3). На данной странице после заполнения всех полей и ввода удовлетворяющего требованиям пароля можно создать учетную запись пользователя.

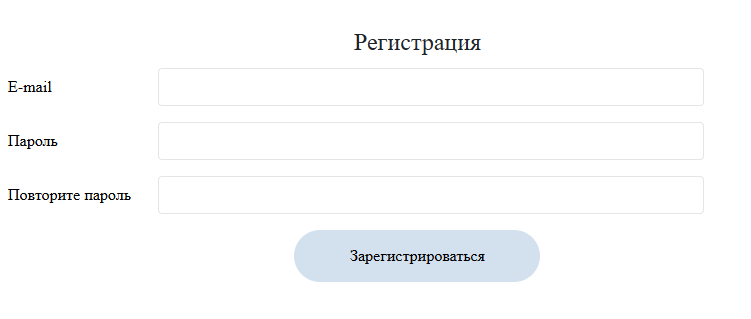


Рисунок 5.3 – Окно регистрации

У зарегистрированного пользователя функционал сайта имеет чуть больше возможностей, а именно добавление нарушений и просмотр уже добавленных с этой учетной записи (рисунок 5.4).



Рисунок 5.4 – Таблица зарегистрированных нарушений

Для того чтобы добавить нарушение необходимо нажать кнопку «Добавить нарушение», после чего откроется соответствующая форма (рисунок 5.5).

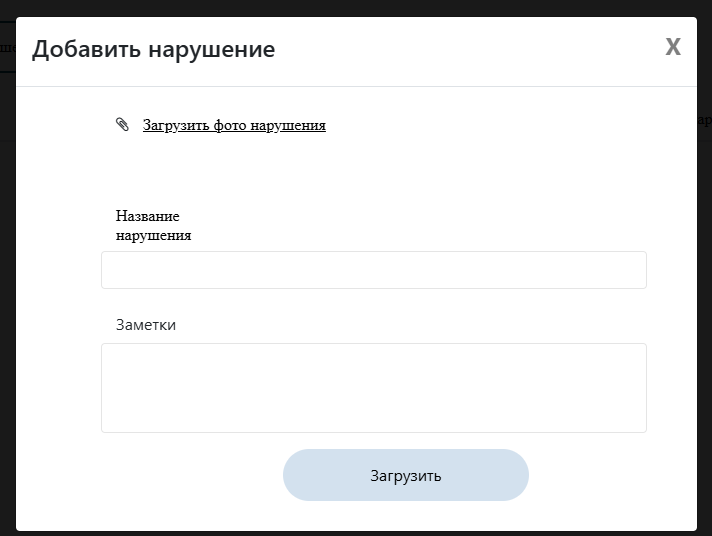


Рисунок 5.5 – Форма добавления нарушения

На данной форме можно загрузить фотографию автомобиля и добавить название нарушения с небольшим описанием. После нажатия кнопки «Загрузить» данное нарушение добавится в соответствующую таблицу и при помощи нейросети будет распознан номер на фотографии (рисунок 5.6).

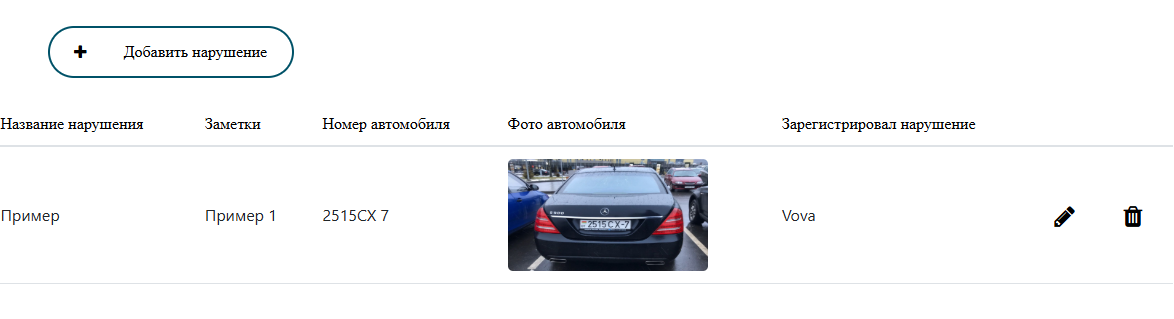


Рисунок 5.6 – Таблица добавленных нарушений

Если пользователь ввел некорректные данные или нейросеть распознало не совсем корректно номер автомобиля, то есть возможность внести корректировки.

Для изменения данных необходимо нажать иконку карандаша, после чего откроется страница где можно будет редактировать внесенные значения (рисунок 5.7). После нажатия кнопки «Загрузить» все эти изменения сохранятся в таблице.

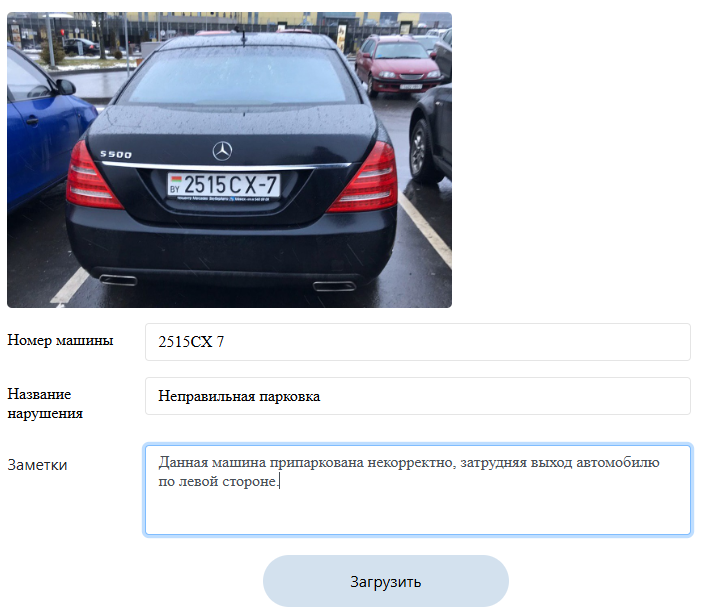


Рисунок 5.7 – Окно редактирования нарушения

Пользователь также может удалить внесенное нарушение. Для этого необходимо нажать соответствующую кнопку, после чего данная информация исчезнет из таблицы.

На главной странице сайта также присутствует возможность поиск нарушений по номеру автомобиля (рисунок 5.8).

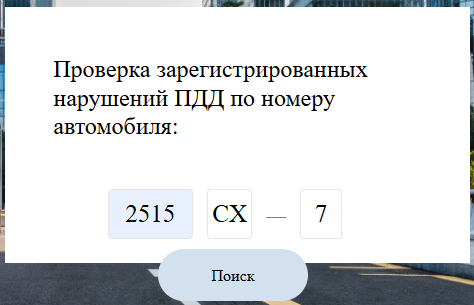


Рисунок 5.8 – Форма поиска

После нажатия кнопки «Поиск» появится таблица с зарегистрированными нарушениями по данному номеру автомобиля (рисунок 5.9) либо сообщение, где будет сказано что по данному номеру нарушений не зафиксировано.

Чтобы пользователю выйти из своей учетной записи необходимо нажать на иконку входа, после чего сайт вернется в исходное состояние.

**5.3 Руководство администратора**

Для того чтобы зайти на сайт в качестве администратора необходимо перейти на страницу входа и ввести соответствующие данные. После авторизации у администратора появляются новые возможности по сравнению с обычными пользователями.

Основное изменение это страница пользователей (рисунок 5.9). На данной странице отображается информация о всех пользователях.

На данном окне есть возможность изменения ролей у пользователей (кроме встроенного администратора). Для этого необходимо выбрать какую роль должен иметь пользователь после чего нажать кнопку «Применить изменения». При следующем заходе измененного пользователя у него уже будут другие возможности.

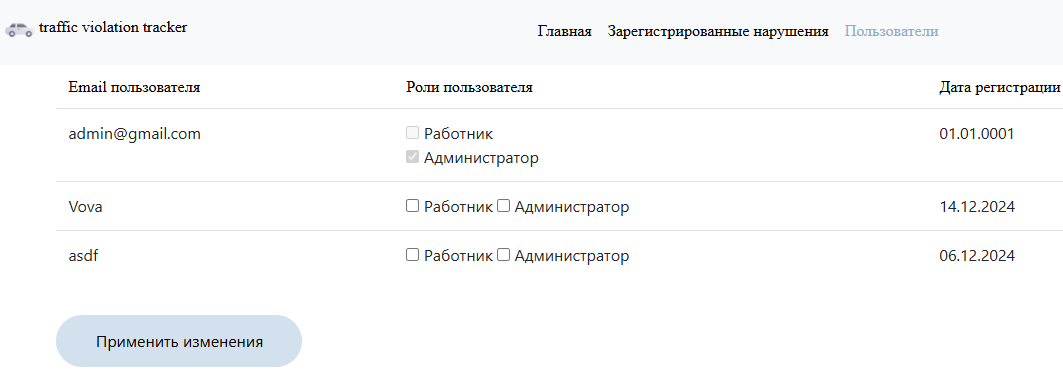


Рисунок 5.9 – Окно пользователей

Также отличие администратора от пользователя в том, что он может просматривать и редактировать все нарушения, внесенные на сайт (рисунок 5.10).

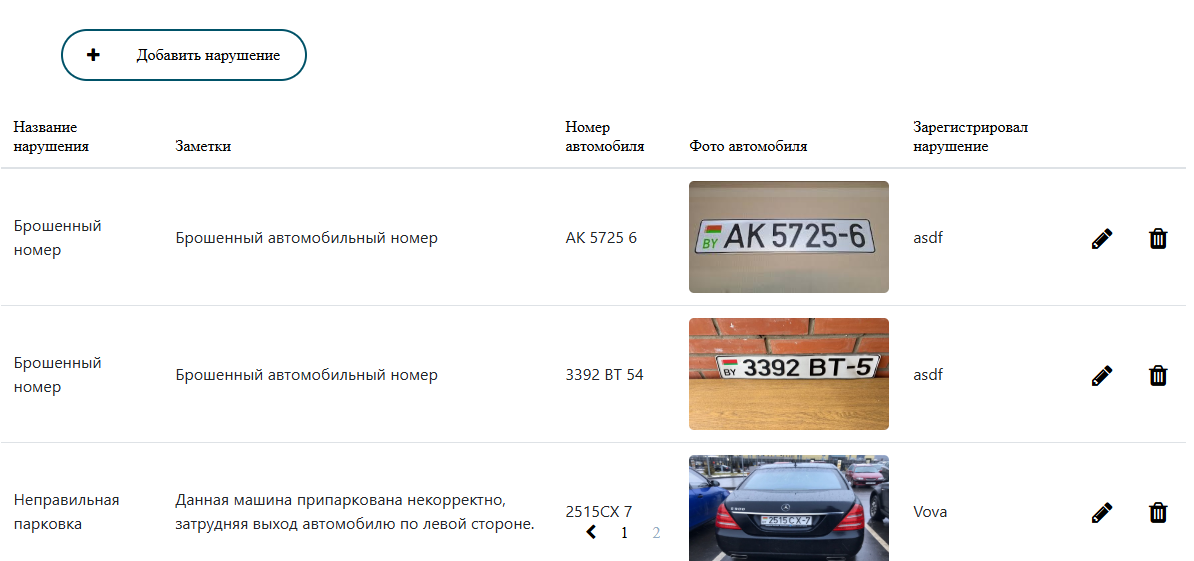


Рисунок 5.10 – Зарегистрированные нарушения

В остальном возможности администратора схожи с теми, что есть у обычного зарегистрированного пользователя.

Для того чтобы выйти с учетной записи администратора необходимо нажать кнопку входа, после чего сайт вернется в исходное состояние.

**6. Технико-экономическое обоснование разработки веб-сервиса распознавания государственных номеров транспортных средств с использованием библиотеки OpenCV**

**6.1 Общая характеристика разрабатываемого веб-приложения**

Разрабатываемое веб-приложение представляет собой комплексное программное решение для автоматического распознавания государственных регистрационных номеров транспортных средств. Оно предназначено для решения задач, связанных с идентификацией транспортных средств, фиксацией нарушений правил дорожного движения, мониторингом транспортных потоков, управлением доступом на парковки и автоматизацией других процессов, требующих анализа данных о транспортных средствах. Основой работы приложения является интеграция с библиотекой OpenCV, которая предоставляет широкий набор инструментов для обработки изображений и видео.

Приложение разработано в виде веб-сервиса, что позволяет обеспечить доступ к его функциям из любой точки с помощью браузера или клиентского приложения. Архитектура сервиса состоит из двух основных частей: клиентской и серверной. Клиентская часть предназначена для взаимодействия с пользователем и представляет собой веб-интерфейс, который предоставляет пользователю удобный доступ ко всем функциям приложения. С помощью этого интерфейса можно загружать изображения или видеопотоки, инициировать процесс распознавания, просматривать результаты обработки, экспортировать данные в различных форматах (например, CSV), а также управлять учетными записями и настройками системы. Клиентская часть разработана с учетом требований удобства использования, что делает её интуитивно понятной даже для неподготовленных пользователей.

Серверная часть приложения выполняет все основные вычислительные операции, включая обработку изображений, распознавание регистрационных номеров, управление данными и взаимодействие с базой данных. Она разработана на основе ASP.NET Core, что обеспечивает высокую производительность, надежность и возможность кросс-платформенного развертывания. Взаимодействие между клиентом и сервером реализовано через REST API, который обеспечивает стандартизированный и безопасный обмен данными. Это решение позволяет легко интегрировать сервис с другими системами или использовать его как компонент в более крупной программной архитектуре.

Ключевым элементом функционала приложения является использование алгоритмов компьютерного зрения, реализованных в библиотеке OpenCV. Эти алгоритмы включают этапы предобработки изображений, такие как удаление шумов, коррекция освещения, нормализация яркости и контраста. После предобработки система выделяет область, содержащую регистрационный номер, выполняет её сегментацию и распознает символы с помощью алгоритмов оптического распознавания символов (OCR). Высокая точность распознавания достигается благодаря использованию оптимизированных алгоритмов и поддержке аппаратного ускорения, что особенно важно при работе с изображениями низкого качества или сложными условиями освещения.

Для хранения данных используется реляционная база данных Microsoft SQL Server, которая обеспечивает надежное хранение информации о результатах распознавания, загруженных изображениях и учетных записях пользователей. Данные хранятся в структурированном виде, что упрощает их последующую обработку, анализ и экспорт. База данных интегрирована с серверной частью приложения через Entity Framework Core, что упрощает работу с данными, обеспечивает поддержку миграций схемы базы данных и позволяет адаптировать систему к изменяющимся требованиям бизнеса.

Одной из ключевых особенностей приложения является его гибкая и масштабируемая архитектура. Она позволяет эффективно обрабатывать большие объемы данных и поддерживать работу с большим количеством пользователей. Для достижения этих целей используется контейнеризация с помощью Docker, что упрощает развертывание приложения, управление зависимостями и переносимость между различными средами выполнения. Это решение делает приложение подходящим для развертывания как в локальных инфраструктурах, так и в облачных платформах, таких как Microsoft Azure или AWS.

Приложение также обладает развитой системой безопасности. Авторизация и аутентификация пользователей реализованы с использованием встроенных средств ASP.NET Core Identity, которые поддерживают хэширование паролей, защиту от атак с подбором паролей и возможность двухфакторной аутентификации. Кроме того, все данные, передаваемые между клиентом и сервером, защищены с использованием протокола HTTPS, что предотвращает их перехват и модификацию.

Интерфейс приложения разработан с учетом современных стандартов пользовательского опыта (UX/UI). Все действия пользователя сопровождаются подсказками, а возможные ошибки ввода данных обрабатываются с выводом детализированных сообщений. Это позволяет минимизировать количество ошибок и делает процесс работы с приложением простым и комфортным.

Разрабатываемое приложение ориентировано на решение широкого круга задач в сфере автоматизации процессов, связанных с регистрацией и идентификацией транспортных средств. Оно предоставляет пользователю мощные инструменты для работы с данными, высокую производительность и надежность, а также гибкость для адаптации под конкретные требования. Благодаря использованию современных технологий, таких как OpenCV, ASP.NET Core и Docker, приложение сочетает в себе передовые подходы к разработке программного обеспечения и надежные решения для реализации бизнес-процессов.

Конкурирующие аналоги: OpenALPR, ANPR Cloud, Sighthound ALPR

Разрабатываемое веб-приложение ориентировано на автоматическое распознавание государственных регистрационных номеров транспортных средств с использованием библиотеки OpenCV. Система предназначена для решения задач в сферах дорожного контроля, управления парковками, мониторинга транспорта и автоматизации бизнес-процессов. После достижения окупаемости и масштабирования проекта планируется добавление поддержки анализа видеопотоков в реальном времени, интеграция с внешними аналитическими платформами и внедрение мобильных приложений для Android и iOS.

Монетизация приложения будет осуществляться через продажу лицензий на использование сервиса, а также предоставление дополнительных услуг, таких как обучение пользователей, настройка системы под специфические требования заказчиков и техническая поддержка. Основной источник прибыли — расширение клиентской базы за счёт повышения точности и производительности распознавания, а также интеграции с системами безопасности и управления.

На основе исследования целевой аудитории было оценено, что потенциальная клиентская база разрабатываемого приложения включает около 10 000 организаций в странах СНГ. Планируемый охват составляет 3-5% этой аудитории, что эквивалентно 300-500 клиентам. При средней стоимости лицензии в 175 белорусских рублей ожидается месячный доход в диапазоне от 60000 до 100000 рублей. Дополнительно, продажи расширенных услуг и интеграционных решений могут увеличить доход компании на 20-30%.

Таким образом, разработка веб-приложения с использованием OpenCV ориентирована на создание конкурентоспособного продукта, способного эффективно решать задачи распознавания регистрационных номеров, автоматизировать процессы и предоставлять клиентам новые возможности для интеграции и аналитики.

В перспективе развитие веб-приложения включает реализацию следующих направлений:

1. Расширение функционала – добавление новых возможностей, таких как интеграция с дополнительными модулями аналитики данных, автоматическая генерация отчетов и поддержка видеопотоков в реальном времени.
2. Оптимизация производительности – улучшение алгоритмов обработки изображений для повышения скорости работы системы, особенно при обработке больших объемов данных или сложных условий (например, ночная съемка или низкое качество изображений).
3. Масштабирование и интеграция – разработка инструментов для интеграции с другими системами, такими как CRM, ERP или системы управления доступом. Это позволит расширить спектр применения веб-приложения.
4. Повышение уровня безопасности – внедрение дополнительных мер защиты данных, таких как шифрование на уровне базы данных, защита от SQL-инъекций и усиление авторизации с помощью многофакторной аутентификации.
5. Локализация интерфейса – добавление поддержки нескольких языков для удобства работы с приложением в международных проектах.
6. Мобильная адаптация – разработка мобильного приложения для взаимодействия с системой, что позволит пользователям получать доступ к функционалу из любого места.

Таким образом, планируемое развитие веб-приложения направлено на повышение его гибкости, функциональности и надежности, что позволит укрепить позиции среди конкурентов и обеспечить успешное применение в различных сферах.

**6.2 Расчет инвестиций в разработку веб-приложения**

Первично необходимо произвести расчет затрат на основную заработную плату. Для данного расчета используется формула 6.1.

(6.1)

,

где – затраты на основную заработную плату, руб.;

– коэффициент премий;

– трудоемкость работ, ч;

– часовой оклад плата исполнителя i-ой категории, руб.

Коэффициент премий заложен в размере учтен в месячном окладе поэтому приравнивается к единице.

Зарплатные медианы специалистов:

* Программист – 5065 руб;
* Тестировщик – 2383,60 руб;
* Дизайнер – 2308 руб;
* Бизнес-аналитик – 5589,14 руб.

Данные были взяты из представленной аналитики вакансий за 2024 год [13][14].

Часовой оклад вычисляется путем деления месячного оклада на 168 (среднемесячное количество рабочих часов).

Трудоемкость работ:

* Программист – 54 ч;
* Тестировщик – 21 ч;
* Дизайнер – 13 ч;
* Бизнес-аналитик – 14 ч.

Результаты расчетов затрат на основную заработную для сотрудников сведены в таблицу 6.1.

Таблица 6.1 – Расчет затрат на основную заработную плату

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Категория исполнителя | Месячный оклад, руб. | Часовой оклад, руб. | Трудоемкость работ, ч | Итого, руб. |
| Программист | 5065 | 30 | 54 | 1620 |
| Тестировщик | 2383,60 | 14,2 | 21 | 298,2 |
| Дизайнер | 2308 | 13,7 | 13 | 178,1 |
| Бизнес-аналитик | 5589,14 | 33,3 | 14 | 466,2 |
| *Итого* | | | | 2562,5 |
| *Премия и иные стимулирующие выплаты* | | | | 1281,2 |
| Всего затрат на основную заработную плату разработчиков | | | | 3843,75 |

Далее необходимо рассчитать дополнительную заработную плату сотрудников. Для этого применяется формула 6.2.

(6.2)

где – затраты на дополнительную заработную плату, руб.;

– затраты на основную заработную плату, руб.;

– норматив дополнительной заработной платы.

Для организации принимается норматив дополнительной заработной платы в размеры 20%. Следовательно, дополнительная заработная плата сотрудников: 3843,75 × 0,2 = 768,75 руб.

После расчета основной и дополнительной заработной платы необходимо рассчитать отчисления на социальные нужды. Для этого используется формула 6.3.

(6.3)

где – отчисления на социальные нужды, руб.;

– затраты на основную заработную плату, руб.;

– затраты на дополнительную заработную плату, руб.;

– норматив отчислений в ФСЗН и Белгосстрах.

Актуальный норматив отчислений равен 34,6%, следовательно, для сотрудников отчисления на социальные нужды, следующие: (3843,75 + 768,75) × 0,346 = 1 595,93 руб.

Далее необходимо рассчитать прочие расходы на разработку приложения по формуле 6.4.

(6.4)

где – прочие расходы, руб.;

– затраты на основную заработную плату, руб.;

– норматив прочих расходов.

принят в размере 35%, следовательно прочие расходы составляют: 3843,75 × 0,35 = 1 345,31 руб.

Осталось рассчитать расходы на реализацию приложения по формуле 6.5.

(6.5)

где – расходы на реализацию, руб.;

– затраты на основную заработную плату, руб.;

– норматив расходов на реализацию.

Норматив расходов на реализацию принят в размере 4%, то прочие расходы: 3843,75 × 0,04 = 153,75 руб.

Для расчета общей суммы затрат результаты были сведены в таблицу 6.2.

Таблица 6.2 – Расчет затрат на разработку приложения

|  |  |  |
| --- | --- | --- |
| Наименование статьи затрат | Формула/таблица для расчета | Значение, руб. |
| Основная заработная плата () | Расчеты представлены в таблице 6.1. | 3843,75 |
| Дополнительная заработная плата ( |  | 768,75 |
| Отчисления на социальные нужды () |  | 1 595,93 |
| Прочие расходы () |  | 1 345,31 |
| Расходы на реализацию (Рр) |  | 153,75 |
| Общая сумма затрат на разработку () |  | 7 707,49 |

По результатам расчетов общий размер инвестиций в разработку веб-приложения равняется 7 707,49 рублей.

**6.3 Расчет экономического эффекта от реализации программного средства на рынке**

Финансовая модель приложения предусматривает монетизацию через продажу лицензий на его использование. Ключевым драйвером доходности проекта выступает расширение числа пользователей, основанное на высокой точности распознавания и производительности приложения, а также его интеграции с внешними системами безопасности и управления.

Согласно результатам анализа целевой аудитории, потенциальный рынок сбыта охватывает около 10 000 организаций на территории стран СНГ. Предполагается, что приложение охватит 3-5% этого рынка, что соответствует 300-500 клиентам. Средняя стоимость лицензии установлена на уровне 175 белорусских рублей.

Проанализировав рынок, рассчитав стоимость продукта и количество покупателей можно перейти к расчету прироста чистой прибыли. Для расчета чистой прибыли используется формула 6.6.

(6.6)

где – прирост чистой прибыли, руб.;

– отпускная цена копии, руб.;

– количество копий, шт.;

– сумма налога на добавочную стоимость, руб.;

– рентабельность продаж, руб.;

– ставка налога на прибыль.

Для разрабатываемого приложения стоимость подписки – 200 руб., приблизительное количество покупок– 420 шт., НДС рассчитывается по формуле 6.7, рентабельность продаж для приложения находится на уровне 25%, ставка налога на прибыль – 20%.

(6.7)

где – сумма налога на добавочную стоимость, руб.;

– стоимость подписок, руб.;

– количество оформляемых в год подписок, шт.;

– ставка налога на добавленную стоимость (20%).

Следовательно сумма налога на добавочную стоимость (НДС) равняется: 175 × 350 × 20% / 120% = 10 207,9 руб.

Определив НДС можно перейти к расчетам прироста чистой прибыли: (175 × 350 – 10 207,9) × 25% × (1 – 0,2) = 10 208,42 руб.

**6.4 Расчет показателей экономической эффективности разработки и реализации веб-сервиса**

Для расчета экономической эффективности разработки необходимо рассчитать рентабельность инвестиций ROI (Return of Investment) для этого используется формула 6.8.

(6.8)

где – рентабельность инвестиций;

– общая сумма затрат на разработку и реализацию, руб.;

– прирост чистой прибыли, руб.

Для разрабатываемого веб-приложения общая сумма затрат на разработку и реализацию равняется 7707,49 руб., а прирост чистой прибыли – 14 351,15 руб.

Следовательно, ROI равняется: (10 208,42 – 7707,49) / 7707,49 = 0,32 = 32%.

Для того чтобы определить рентабельность приложения необходимо сравнить рентабельность инвестиций ROI с рентабельностью долгосрочного вклада той же суммы.

На ноябрь 2024 года в Республике Беларусь самая высокая ставка по долгосрочным депозитам по результатам анализа предложений банков Республики Беларусь и информации с сайта MYFIN (представлена на рисунке 6.2) равняется – 14,5% [15].

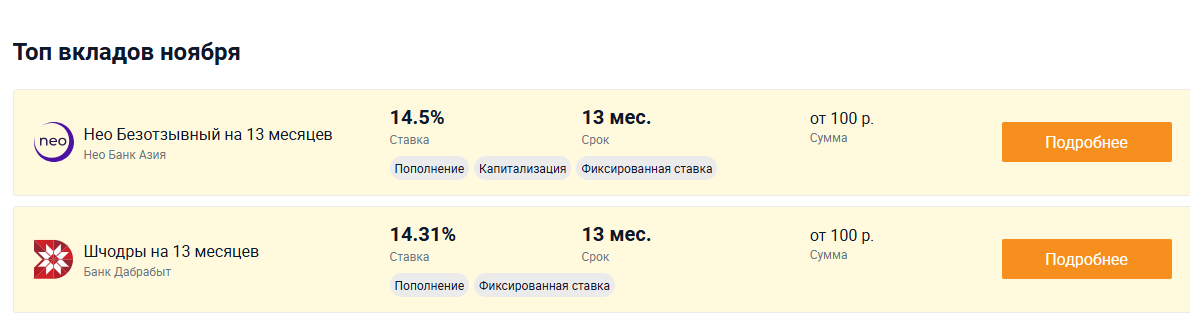


Рисунок 6.1 – Информация о ставках долгосрочных депозитов.

Ставка в размере 14,5% меньше полученного значения рентабельности инвестиций (32%), следовательно, разработка данного приложения является экономически эффективной и принесёт организации существенную прибыль.

## **Заключение**

В ходе выполнения дипломного проекта был разработан веб-сервис для распознавания государственных регистрационных номеров транспортных средств, направленный на автоматизацию процессов идентификации автомобилей в различных сферах. В ходе работы были изучены современные методы компьютерного зрения, алгоритмы оптического распознавания символов и технологии обработки изображений, которые легли в основу создания серверной и клиентской архитектуры веб-сервиса. Это позволило реализовать надежную систему для точного и быстрого распознавания регистрационных номеров в реальном времени.

Основной целью данного проекта являлось создание программного решения, обеспечивающего высокую точность распознавания и удобство управления результатами. Был разработан веб-сервис, который не только поддерживает базовые функции обработки изображений и их анализа, но и предлагает гибкое управление данными и пользователями системы. В процессе разработки были реализованы ключевые функции, включая загрузку и предобработку изображений, распознавание номерных знаков, управление результатами, а также поддержка ролевого доступа и авторизации. Особое внимание было уделено обеспечению безопасности и защиты данных, что особенно важно для работы с конфиденциальной информацией, связанной с транспортными средствами.

Тестирование разработанного сервиса показало его высокую эффективность и стабильность работы в различных условиях, включая низкое качество изображений и изменчивые параметры освещения. Это подтвердило, что созданное программное решение способно успешно решать задачи автоматической идентификации регистрационных номеров транспортных средств, а также предоставлять пользователям удобные инструменты для анализа и управления данными. Внедрение системы в реальных условиях позволяет значительно повысить точность, скорость и эффективность процессов контроля и мониторинга транспортных средств.

Практическая значимость выполненной работы заключается в создании инструмента, который может быть интегрирован в существующие системы безопасности, системы управления дорожным движением, а также автоматизированные парковочные комплексы. Полученные результаты могут быть использованы для дальнейшей разработки и масштабирования системы, внедрения в смежные сферы и адаптации под различные требования заказчиков.

Таким образом, результаты дипломного проектирования подтвердили, что внедрение технологий распознавания номерных знаков способствует значительному улучшению процессов управления транспортными средствами. Созданное программное решение демонстрирует высокий уровень автоматизации и надежности, что особенно важно для обеспечения безопасности и оптимизации работы в различных сферах, требующих автоматической идентификации транспортных средств.

# **Список использованных источников**

1. Кнут, Д. Искусство программирования. Т.1: Основные алгоритмы. — М.: Вильямс, 2016. — 736 с.
2. Рихтер, Д. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. — СПб.: Питер, 2015. — 896 с.
3. Альбахари, Дж., Альбахари, Б. C# 9.0 in a Nutshell: The Definitive Reference. — O'Reilly Media, 2021. — 1088 с.
4. Pro ASP.NET Core MVC 2 / Адам Фримен. — Apress, 2017. — 1017 с.
5. Фаулер, М. Архитектура корпоративных программных приложений. Паттерны проектирования. — М.: Вильямс, 2011. — 560 с.
6. Docker: Up & Running / Фишер, К., Готтфрид, И. — O'Reilly Media, 2020. — 350 с.
7. Microsoft SQL Server 2019: A Beginner’s Guide / Мартинес, Р. — McGraw-Hill Education, 2020. — 720 с.
8. OpenCV 4 with Python Blueprints / Хурана, В. — Packt Publishing, 2019. — 350 с.
9. Основы REST API для веб-приложений / Хэдс, Д. — O'Reilly Media, 2018. — 280 с.
10. Современные подходы к разработке баз данных / Иванов, А. — М.: Вильямс, 2017. — 430 с.
11. Различные виды тестирования ПО [Электронный ресурс]. – Режим доступа: <https://www.atlassian.com/ru/continuous-delivery/software-testing/types-of-software-testing>.
12. Тест план. Тест кейс. Чек лист [Электронный ресурс]. – Режим доступа: <https://qalearning.net/home/usefullresourses/testplan_checklist_testcase>.
13. Зарплата в IT в Минске [Электронный ресурс]. – Режим доступа: https://wadline.ru/salary/minsk.
14. Зарплаты в IT в Беларуси [Электронный ресурс]. – Режим доступа: https://salaries.devby.io/.
15. Долгосрочные вклады в банках [Электронный ресурс]. – Режим доступа: https://myfin.by/vklady.

# **Приложение А**

(обязательное)

Исходный текст программы

using Microsoft.AspNetCore.Hosting;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.Hosting;

using Microsoft.Extensions.Logging;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Identity;

using Microsoft.Extensions.DependencyInjection;

using WebAppCars.Models;

namespace WebAppCars

{

public class Program

{

public static async Task Main(string[] args)

{

var host = CreateHostBuilder(args).Build();

using (var scope = host.Services.CreateScope())

{

var services = scope.ServiceProvider;

try

{

var userManager = services.GetRequiredService<UserManager<UserCompany>>();

var rolesManager = services.GetRequiredService<RoleManager<IdentityRole>>();

await RoleInitializer.InitializeAsync(userManager, rolesManager);

}

catch (Exception ex)

{

var logger = services.GetRequiredService<ILogger<Program>>();

logger.LogError(ex, "An error occurred while seeding the database.");

}

}

host.Run();

}

public static IHostBuilder CreateHostBuilder(string[] args) =>

Host.CreateDefaultBuilder(args)

.ConfigureWebHostDefaults(webBuilder =>

{

webBuilder.UseStartup<Startup>();

});

}

}

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Identity;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.EntityFrameworkCore;

using WebAppCars.Models;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Diagnostics;

using System.Net;

using System.IO;

namespace WebAppCars

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

public IConfiguration Configuration { get; }

public void ConfigureServices(IServiceCollection services)

{

services.AddDbContext<ApplicationContext>(options =>

options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection")));

services.AddIdentity<UserCompany, IdentityRole>()

.AddEntityFrameworkStores<ApplicationContext>()

.AddDefaultTokenProviders();

services.AddControllersWithViews();

}

public void Configure(IApplicationBuilder app)

{

app.UseStatusCodePagesWithReExecute("/StatusCode/{0}");

app.UseHsts();

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthentication();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}");

});

}

}

}

using WebAppCars.Models;

using Microsoft.AspNetCore.Identity;

using System.Threading.Tasks;

namespace WebAppCars

{

public class RoleInitializer

{

public static async Task InitializeAsync(UserManager<UserCompany> userManager, RoleManager<IdentityRole> roleManager)

{

string adminEmail = "admin@gmail.com";

string password = "\_Aa123456";

if (await roleManager.FindByNameAsync("admin") == null)

{

await roleManager.CreateAsync(new IdentityRole("admin"));

}

if (await roleManager.FindByNameAsync("employee") == null)

{

await roleManager.CreateAsync(new IdentityRole("employee"));

}

if (await userManager.FindByNameAsync(adminEmail) == null)

{

UserCompany admin = new UserCompany { Email = adminEmail, UserName = adminEmail };

IdentityResult result = await userManager.CreateAsync(admin, password);

if (result.Succeeded)

{

await userManager.AddToRoleAsync(admin, "admin");

}

}

}

}

}

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Identity;

using Microsoft.AspNetCore.Identity.EntityFrameworkCore;

using WebAppCars.Models;

using WebAppCars.ViewModels;

using Microsoft.AspNetCore.Authorization;

namespace WebAppCars.Controllers

{

[Authorize(Roles = "admin")]

public class RolesController : Controller

{

RoleManager<IdentityRole> \_roleManager;

UserManager<UserCompany> \_userManager;

public RolesController(RoleManager<IdentityRole> roleManager, UserManager<UserCompany> userManager)

{

\_roleManager = roleManager;

\_userManager = userManager;

}

public async Task<IActionResult> Index()

{

var users = new List<UserRoleViewModel>();

foreach (var user in \_userManager.Users.ToList())

{

var userRoles = await \_userManager.GetRolesAsync(user);

var model = new UserRoleViewModel

{

UserMail = user.Email,

UserRoles = userRoles,

UserRegistrationDate = user.RegistrationDate,

};

users.Add(model);

}

if (users.Count > 0)

{

return View(users);

}

return NotFound();

}

public IActionResult Create() => View();

[HttpPost]

public async Task<IActionResult> Create(string name)

{

if (!string.IsNullOrEmpty(name))

{

IdentityResult result = await \_roleManager.CreateAsync(new IdentityRole(name));

if (result.Succeeded)

{

return RedirectToAction("Index");

}

else

{

foreach (var error in result.Errors)

{

ModelState.AddModelError(string.Empty, error.Description);

}

}

}

return View(name);

}

[HttpPost]

public async Task<IActionResult> Delete(string id)

{

IdentityRole role = await \_roleManager.FindByIdAsync(id);

if (role != null)

{

IdentityResult result = await \_roleManager.DeleteAsync(role);

}

return RedirectToAction("Index");

}

public IActionResult UserList() => View(\_userManager.Users.ToList());

public async Task<IActionResult> Edit(string userId)

{

UserCompany user = await \_userManager.FindByIdAsync(userId);

if (user != null)

{

var userRoles = await \_userManager.GetRolesAsync(user);

var allRoles = \_roleManager.Roles.ToList();

ChangeRoleViewModel model = new ChangeRoleViewModel

{

UserId = user.Id,

UserEmail = user.Email,

UserRoles = userRoles,

AllRoles = allRoles

};

return View(model);

}

return NotFound();

}

[HttpPost]

public async Task<IActionResult> Edit(string userId, List<string> roles)

{

UserCompany user = await \_userManager.FindByIdAsync(userId);

if (user != null)

{

var userRoles = await \_userManager.GetRolesAsync(user);

var allRoles = \_roleManager.Roles.ToList();

var addedRoles = roles.Except(userRoles);

var removedRoles = userRoles.Except(roles);

await \_userManager.AddToRolesAsync(user, addedRoles);

await \_userManager.RemoveFromRolesAsync(user, removedRoles);

return RedirectToAction("UserList");

}

return NotFound();

}

[HttpPost]

public async Task<IActionResult> Remember(IEnumerable<UserRoleViewModel> userModels)

{

foreach (var item in userModels)

{

var user = await \_userManager.FindByEmailAsync(item.UserMail);

if (user != null)

{

if (item.UserRoles == null)

{

item.UserRoles = new List<string>();

}

var userRoles = await \_userManager.GetRolesAsync(user);

var addedRoles = item.UserRoles.Except(userRoles);

var removedRoles = userRoles.Except(item.UserRoles);

await \_userManager.AddToRolesAsync(user, addedRoles);

await \_userManager.RemoveFromRolesAsync(user, removedRoles);

}

}

return RedirectToAction("Index");

}

}

}

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Identity;

using WebAppCars.Models;

using WebAppCars.ViewModels;

namespace WebAppCars.Controllers

{

public class UsersController : Controller

{

UserManager<UserCompany> \_userManager;

public UsersController(UserManager<UserCompany> userManager)

{

\_userManager = userManager;

}

public IActionResult Index() => View(\_userManager.Users.ToList());

public IActionResult Create() => View();

[HttpPost]

public async Task<IActionResult> Create(CreateUserViewModel model)

{

if (ModelState.IsValid)

{

UserCompany user = new UserCompany { Email = model.Email, UserName = model.Email, LeadName = model.LeadName, CompanyName = model.CompanyName, RegistrationDate = DateTime.Now };

var result = await \_userManager.CreateAsync(user, model.Password);

if (result.Succeeded)

{

return RedirectToAction("Index");

}

else

{

foreach (var error in result.Errors)

{

ModelState.AddModelError(string.Empty, error.Description);

}

}

}

return View(model);

}

public async Task<IActionResult> Edit(string id)

{

UserCompany user = await \_userManager.FindByIdAsync(id);

if (user == null)

{

return NotFound();

}

EditUserViewModel model = new EditUserViewModel { Id = user.Id, Email = user.Email, LeadName = user.LeadName, CompanyName = user.CompanyName };

return View(model);

}

[HttpPost]

public async Task<IActionResult> Edit(EditUserViewModel model)

{

if (ModelState.IsValid)

{

UserCompany user = await \_userManager.FindByIdAsync(model.Id);

if (user != null)

{

user.Email = model.Email;

user.UserName = model.Email;

user.LeadName = model.LeadName;

user.CompanyName = model.CompanyName;

var result = await \_userManager.UpdateAsync(user);

if (result.Succeeded)

{

return RedirectToAction("Index");

}

else

{

foreach (var error in result.Errors)

{

ModelState.AddModelError(string.Empty, error.Description);

}

}

}

}

return View(model);

}

[HttpPost]

public async Task<ActionResult> Delete(string id)

{

UserCompany user = await \_userManager.FindByIdAsync(id);

if (user != null)

{

IdentityResult result = await \_userManager.DeleteAsync(user);

}

return RedirectToAction("Index");

}

[HttpPost]

public async Task<IActionResult> ChangePassword(ChangePasswordViewModel model)

{

if (ModelState.IsValid)

{

UserCompany user = await \_userManager.FindByIdAsync(model.Id);

if (user != null)

{

IdentityResult result =

await \_userManager.ChangePasswordAsync(user, model.OldPassword, model.NewPassword);

if (result.Succeeded)

{

return RedirectToAction("Index");

}

else

{

foreach (var error in result.Errors)

{

ModelState.AddModelError(string.Empty, error.Description);

}

}

}

else

{

ModelState.AddModelError(string.Empty, "Пользователь не найден");

}

}

return View(model);

}

}

}

using Emgu.CV;

using Emgu.CV.CvEnum;

using Emgu.CV.OCR;

using Emgu.CV.Structure;

using Emgu.CV.Util;

using Emgu.Util;

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Drawing.Imaging;

using System.IO;

using System.Linq;

using System.Text;

namespace Recognizer

{

public class NumberPlateRecognizer : DisposableObject

{

private Tesseract \_ocr;

public List<byte[]> plates1 = new List<byte[]>();

public NumberPlateRecognizer(string language)

{

var temp = new DirectoryInfo(Environment.CurrentDirectory + @"../../../tessdata").FullName;

\_ocr = new Tesseract(temp, language, OcrEngineMode.TesseractOnly);

\_ocr.SetVariable("tessedit\_char\_blacklist", "¢§+~»~`!@#$%^&\*()\_+-={}[]|\\:\";\'<>?,./абвгдеёжзийклмнопрстуфхцчшщъыьэюяАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ");

}

public List<string> ProcessImage(Bitmap image)

{

List<IInputOutputArray> licensePlateImageList = new List<IInputOutputArray>();

List<IInputOutputArray> filteredLicensePlatImageList = new List<IInputOutputArray>();

List<RotatedRect> licenseBoxList = new List<RotatedRect>();

Rectangle rectangle = new Rectangle(0, 0, image.Width, image.Height);

BitmapData bmpData = image.LockBits(rectangle, ImageLockMode.ReadWrite, image.PixelFormat);

Image<Bgr, byte> outputImage = new Image<Bgr, byte>(image.Width, image.Height, bmpData.Stride, bmpData.Scan0);

List<string> recognizedPlates = this.DetectLicensePlate(outputImage,

licensePlateImageList, filteredLicensePlatImageList,

licenseBoxList);

foreach (var item in licensePlateImageList)

{

plates1.Add(item.GetInputArray().GetUMat().ToImage<Bgr, byte>().ToJpegData());

}

return recognizedPlates;

}

private List<String> DetectLicensePlate(

IInputArray img,

List<IInputOutputArray> licensePlateImagesList,

List<IInputOutputArray> filteredLicensePlateImagesList,

List<RotatedRect> detectedLicensePlateRegionList)

{

List<String> licenses = new List<String>();

using (Mat gray = new Mat())

using (Mat gauss = new Mat())

using (Mat canny = new Mat())

using (VectorOfVectorOfPoint contours = new VectorOfVectorOfPoint())

{

CvInvoke.CvtColor(img, gray, ColorConversion.Bgr2Gray);

CvInvoke.GaussianBlur(gray, gauss, new Size(11, 11), 0.8, borderType: BorderType.Wrap);

CvInvoke.Canny(gauss, canny, 100, 50, 3, false);

int[,] hierachy = CvInvoke.FindContourTree(canny, contours, ChainApproxMethod.ChainApproxSimple);

FindLicensePlate(contours, hierachy, 0, gray, canny, licensePlateImagesList, filteredLicensePlateImagesList, detectedLicensePlateRegionList, licenses);

}

return licenses;

}

private static int GetNumberOfChildren(int[,] hierachy, int idx)

{

//first child

idx = hierachy[idx, 2];

if (idx < 0)

return 0;

int count = 1;

while (hierachy[idx, 0] > 0)

{

count++;

idx = hierachy[idx, 0];

}

return count;

}

private void FindLicensePlate(

VectorOfVectorOfPoint contours, int[,] hierachy, int idx, IInputArray gray, IInputArray canny,

List<IInputOutputArray> licensePlateImagesList, List<IInputOutputArray> filteredLicensePlateImagesList, List<RotatedRect> detectedLicensePlateRegionList,

List<String> licenses)

{

for (; idx >= 0; idx = hierachy[idx, 0])

{

int numberOfChildren = GetNumberOfChildren(hierachy, idx);

//if it does not contains any children (charactor), it is not a license plate region

if (numberOfChildren == 0) continue;

using (VectorOfPoint contour = contours[idx])

{

if (CvInvoke.ContourArea(contour) > 400)

{

if (numberOfChildren < 3)

{

//If the contour has less than 3 children, it is not a license plate (assuming license plate has at least 3 charactor)

//However we should search the children of this contour to see if any of them is a license plate

FindLicensePlate(contours, hierachy, hierachy[idx, 2], gray, canny, licensePlateImagesList,

filteredLicensePlateImagesList, detectedLicensePlateRegionList, licenses);

continue;

}

RotatedRect box = CvInvoke.MinAreaRect(contour);

if (box.Angle < -45.0)

{

float tmp = box.Size.Width;

box.Size.Width = box.Size.Height;

box.Size.Height = tmp;

box.Angle += 90.0f;

}

else if (box.Angle > 45.0)

{

float tmp = box.Size.Width;

box.Size.Width = box.Size.Height;

box.Size.Height = tmp;

box.Angle -= 90.0f;

}

double whRatio = (double)box.Size.Width / box.Size.Height;

if (!(3.0 < whRatio && whRatio < 10.0))

//if (!(1.0 < whRatio && whRatio < 2.0))

{

//if the width height ratio is not in the specific range,it is not a license plate

//However we should search the children of this contour to see if any of them is a license plate

//Contour<Point> child = contours.VNext;

if (hierachy[idx, 2] > 0)

FindLicensePlate(contours, hierachy, hierachy[idx, 2], gray, canny, licensePlateImagesList,

filteredLicensePlateImagesList, detectedLicensePlateRegionList, licenses);

continue;

}

using (UMat tmp1 = new UMat())

using (UMat tmp2 = new UMat())

{

PointF[] srcCorners = box.GetVertices();

PointF[] destCorners = new PointF[] {

new PointF(0, box.Size.Height - 1),

new PointF(0, 0),

new PointF(box.Size.Width - 1, 0),

new PointF(box.Size.Width - 1, box.Size.Height - 1)};

using (Mat rot = CvInvoke.GetAffineTransform(srcCorners, destCorners))

{

CvInvoke.WarpAffine(gray, tmp1, rot, Size.Round(box.Size));

}

//resize the license plate such that the front is ~ 10-12. This size of front results in better accuracy from tesseract

Size approxSize = new Size(240, 180);

double scale = Math.Min(approxSize.Width / box.Size.Width, approxSize.Height / box.Size.Height);

Size newSize = new Size((int)Math.Round(box.Size.Width \* scale), (int)Math.Round(box.Size.Height \* scale));

CvInvoke.Resize(tmp1, tmp2, newSize, 0, 0, Inter.Cubic);

//removes some pixels from the edge

int edgePixelSize = 3;

Rectangle newRoi = new Rectangle(new Point(edgePixelSize, edgePixelSize),

tmp2.Size - new Size(2 \* edgePixelSize, 2 \* edgePixelSize));

UMat plate = new UMat(tmp2, newRoi);

UMat filteredPlate = FilterPlate(plate);

Tesseract.Character[] words;

StringBuilder strBuilder = new StringBuilder();

using (UMat tmp = filteredPlate.Clone())

{

\_ocr.SetImage(tmp);

\_ocr.Recognize();

//strBuilder.Append(\_ocr.GetUTF8Text());

words = \_ocr.GetCharacters();

if (words.Length == 0) continue;

for (int i = 0; i < words.Length; i++)

{

strBuilder.Append(words[i].Text);

}

}

licenses.Add(strBuilder.ToString());

licensePlateImagesList.Add(plate);

filteredLicensePlateImagesList.Add(filteredPlate);

detectedLicensePlateRegionList.Add(box);

}

}

}

}

}

/// <summary>

/// Filter the license plate to remove noise

/// </summary>

/// <param name="plate">The license plate image</param>

/// <returns>License plate image without the noise</returns>

private static UMat FilterPlate(UMat plate)

{

UMat thresh = new UMat();

CvInvoke.Threshold(plate, thresh, 120, 255, ThresholdType.BinaryInv);

//Image<Gray, Byte> thresh = plate.ThresholdBinaryInv(new Gray(120), new Gray(255));

Size plateSize = plate.Size;

using (Mat plateMask = new Mat(plateSize.Height, plateSize.Width, DepthType.Cv8U, 1))

using (Mat plateCanny = new Mat())

using (VectorOfVectorOfPoint contours = new VectorOfVectorOfPoint())

{

plateMask.SetTo(new MCvScalar(255.0));

CvInvoke.Canny(plate, plateCanny, 100, 50);

CvInvoke.FindContours(plateCanny, contours, null, RetrType.External, ChainApproxMethod.ChainApproxSimple);

int count = contours.Size;

for (int i = 0; i < count; i++)

{

using (VectorOfPoint contour = contours[i])

{

Rectangle rect = CvInvoke.BoundingRectangle(contour);

if (rect.Height > (plateSize.Height >> 1))

{

rect.X -= 1; rect.Y -= 1; rect.Width += 2; rect.Height += 2;

Rectangle roi = new Rectangle(Point.Empty, plate.Size);

rect.Intersect(roi);

CvInvoke.Rectangle(plateMask, rect, new MCvScalar(), -1);

//plateMask.Draw(rect, new Gray(0.0), -1);

}

}

}

thresh.SetTo(new MCvScalar(), plateMask);

}

CvInvoke.Erode(thresh, thresh, null, new Point(-1, -1), 1, BorderType.Constant, CvInvoke.MorphologyDefaultBorderValue);

CvInvoke.Dilate(thresh, thresh, null, new Point(-1, -1), 1, BorderType.Constant, CvInvoke.MorphologyDefaultBorderValue);

return thresh;

}

protected override void DisposeObject()

{

\_ocr.Dispose();

}

}

}

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Identity;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Rendering;

using Recognizer;

using System;

using System.Collections.Generic;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using WebAppCars.Models;

using WebAppCars.ViewModels;

using System.Text.RegularExpressions;

namespace WebAppCars.Controllers

{

public class CarSignController : Controller

{

private NumberPlateRecognizer \_recognizer;

ApplicationContext \_context;

private readonly UserManager<UserCompany> \_userManager;

public CarSignController(UserManager<UserCompany> userManager, ApplicationContext context)

{

\_context = context;

\_userManager = userManager;

}

[HttpPost]

public async Task<IActionResult> Recognize(PlateViewModel pvm)

{

UserCompany applicationUser = await \_userManager.GetUserAsync(User);

string language = string.Empty;

switch (pvm.Region)

{

case "Belarus":

language = "bel";

break;

case "Germany":

language = "deu";

break;

case "Great Britain":

language = "eng";

break;

case "Russia":

language = "rus";

break;

}

\_recognizer = new NumberPlateRecognizer(language);

if (pvm.Image != null)

{

byte[] imageData = null;

using (var binaryReader = new BinaryReader(pvm.Image.OpenReadStream()))

{

imageData = binaryReader.ReadBytes((int)pvm.Image.Length);

var image = GetImageFromByteArray(imageData);

image.Save("TestImage.jpg");

var results = \_recognizer.ProcessImage(image);

var result = results.FirstOrDefault();

if (results.Count == 0)

{

TempData["notice"] = "Cannot recognize this image";

return RedirectToAction("Index", "LicensePlate");

}

foreach (var item in results)

{

var numPlate = item.Trim('"', '\'', ' ');

var regex = new Regex(@"\b(\d{4}[^\\d\\W]{2}\s\d{1}|[^\\d\\W]{2}\s\d{4}\s[^\\d\\W]{1}|\d{4}\s[^\\d\\W]{2}[-\s]\d{1}|[^\\d\\W]{2}\s\d{4}[-\s]\d{1})\b");

var match = regex.Match(numPlate);

if (match.Success)

{

result = match.Value;

}

}

var temp = \_recognizer.plates1;

var image1 = GetImageFromByteArray(temp.FirstOrDefault());

image1.Save("temp.jpeg");

var plate = new NumberPlate();

plate.Country = pvm.Region;

plate.Image = temp.FirstOrDefault();

plate.ImageMimeType = pvm.Image.ContentType;

plate.SourceImage = imageData;

plate.ImageMimeType = pvm.Image.ContentType;

plate.PublisherId = applicationUser.Id;

plate.Summary = pvm.Summary;

plate.SignValue = result;

plate.Notes = pvm.Notes;

\_context.Plates.Add(plate);

\_context.SaveChanges();

System.IO.File.Delete("~\\..\\TestImage.jpg");

System.IO.File.Delete("~\\..\\temp.jpeg");

}

}

return RedirectToAction("Index", "LicensePlate");

}

private static Bitmap GetImageFromByteArray(byte[] byteArray)

{

Bitmap bm = new ImageConverter().ConvertFrom(byteArray) as Bitmap;

if (bm != null && (bm.HorizontalResolution != (int)bm.HorizontalResolution ||

bm.VerticalResolution != (int)bm.VerticalResolution))

{

bm.SetResolution((int)(bm.HorizontalResolution + 0.5f),

(int)(bm.VerticalResolution + 0.5f));

}

return bm;

}

}

}

.input\_\_wrapper {

width: 100%;

position: relative;

margin: 15px 0;

}

.input\_\_file {

opacity: 0;

visibility: hidden;

position: absolute;

}

.input-span{

text-decoration:underline;

}

.input-span :hover {

font-size:17px !important;

}

.input\_\_file-icon-wrapper {

height: 60px;

width: 60px;

margin-right: 15px;

display: -webkit-box;

display: -ms-flexbox;

display: flex;

-webkit-box-align: center;

-ms-flex-align: center;

align-items: center;

-webkit-box-pack: center;

-ms-flex-pack: center;

justify-content: center;

border-right: 1px solid #fff;

}

.input\_\_file-button-text {

line-height: 1;

margin-top: 1px;

}

.input\_\_file-button {

width: 100%;

max-width: 290px;

height: 60px;

background: #1bbc9b;

color: #fff;

font-size: 1.125rem;

font-weight: 700;

display: -webkit-box;

display: -ms-flexbox;

display: flex;

-webkit-box-align: center;

-ms-flex-align: center;

align-items: center;

-webkit-box-pack: start;

-ms-flex-pack: start;

justify-content: flex-start;

border-radius: 3px;

cursor: pointer;

margin: 0 auto;

}

using Microsoft.AspNetCore.Identity;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Rendering;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using WebAppCars.Models;

using WebAppCars.ViewModels;

namespace WebAppCars.Controllers

{

public class LicensePlateController : Controller

{

ApplicationContext \_context;

private readonly UserManager<UserCompany> \_userManager;

public LicensePlateController(ApplicationContext context, UserManager<UserCompany> userManager)

{

\_context = context;

\_userManager = userManager;

}

public async Task<IActionResult> Index(int page=1)

{

int pageSize = 4;

UserCompany applicationUser = await \_userManager.GetUserAsync(User);

var plates = \_context.Plates.ToList();

var p = plates.Skip((page - 1) \* pageSize).Take(pageSize).ToList();

if (!User.IsInRole("admin"))

{

plates = plates.Where(x => x.PublisherId == applicationUser.Id).ToList();

p = plates.Skip((page - 1) \* pageSize).Take(pageSize).ToList();

}

var platesVm = new List<NumberPlateVm>();

foreach (var item in p)

{

var itemVm = new NumberPlateVm();

itemVm.Id = item.Id;

itemVm.Country = item.Country;

itemVm.Image = item.Image;

itemVm.ImageMimeType = item.ImageMimeType;

itemVm.Notes = item.Notes;

itemVm.PublisherName = \_context.Users.Where(x => x.Id == item.PublisherId).FirstOrDefault().UserName;

itemVm.Summary = item.Summary;

itemVm.SourceImage = item.SourceImage;

itemVm.SourceImageMimeType = item.SourceImageMimeType;

itemVm.SignValue = item.SignValue;

platesVm.Add(itemVm);

}

var pageInfo = new LicensePlatePageInfo { PageNumber = page, PageSize = pageSize, TotalItems = plates.Count };

var lpvm = new LicensePlateViewModel { PageInfo = pageInfo, Plates = platesVm };

var regions = new List<string>()

{

"Belarus",

};

ViewBag.Regions = new SelectList(regions);

return View(lpvm);

}

public FileContentResult GetImage(int plateId)

{

var plate = \_context.Plates

.FirstOrDefault(g => g.Id == plateId);

if (plate != null)

{

return File(plate.Image, plate.ImageMimeType);

}

else

{

return null;

}

}

public FileContentResult GetSourceImage(int plateId)

{

var plate = \_context.Plates

.FirstOrDefault(g => g.Id == plateId);

if (plate != null)

{

return File(plate.SourceImage, plate.ImageMimeType);

}

else

{

return null;

}

}

public IActionResult Edit(int id)

{

var itemToEdit = \_context.Plates.Find(id);

var model = new EditPlateViewModel()

{

Id = id,

SignValue = itemToEdit.SignValue,

Summary = itemToEdit.Summary,

Notes = itemToEdit.Notes

};

return View(model);

}

public IActionResult EditPlate(EditPlateViewModel model)

{

var itemToEdit = \_context.Plates.Find(model.Id);

if (itemToEdit != null)

{

itemToEdit.Notes = model.Notes;

itemToEdit.Summary = model.Summary;

itemToEdit.SignValue = model.SignValue;

\_context.SaveChanges();

}

return RedirectToAction("Index");

}

public IActionResult Delete(int id)

{

var itemToDelete = \_context.Plates.Find(id);

if(itemToDelete != null)

{

\_context.Remove(itemToDelete);

\_context.SaveChanges();

}

return RedirectToAction("Index");

}

}

}

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | *Обозначение* | *Наименование* | *Примечание* | |  | *Текстовые документы* |  | |  |  |  | | *БГУИР ДП 1-40 01 01 01 148 ПЗ* | *Пояснительная записка* | *89 с* | |  | *Отзыв руководителя* |  | |  | *Рецензия* |  | |  | *Отчёт о проверке на заимствования* |  | |  |  |  | |  | *Графические документы* |  | |  |  |  | | *ГУИР.181071.001 ПЛ* | *Диаграмма вариантов использования* | *Формат А1* | |  | *Плакат* |  | | *ГУИР.181071.002 ПЛ* | *Диаграмма деятельности* | *Формат А1* | |  | *Плакат* |  | | *ГУИР.181071.003 ПЛ* | *Диаграмма классов программного* | *Формат А1* | |  | *средства* |  | |  | *Плакат* |  | |  |  |  | | *ГУИР.181071.004 СА* | *Добавление нарушения* | *Формат А1* | |  | *Схема алгоритма* |  | | *ГУИР.181071.005 СА* | *Авторизация* | *Формат А1* | |  | *Схема алгоритма* |  | | *ГУИР.181071.006 СА* | *Поиск* | *Формат А1* | |  | *Схема алгоритма* |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | | | | | | | | | | | |
|  |  |  |  |  | *БГУИР ДП 1-40 01 01 01 148 Д1* | | | | | |
|  |  |  |  |  |
| *Изм* | *Лист* | *№ докум.* | *Подп.* | *Дата* |
| *Разраб.* | | *Фролов В.В* |  |  | *Веб-сервис распознавания государственных номеров транспортных средств с использованием библиотеки OpenCV*  *Ведомость дипломного проекта* | *Лит.* | | | *Лист* | *Листов* |
| *Пров.* | | *Сицко А.Л.* |  |  |  | *T* |  | *90* | *90* |
| *Т.контр.* | | *Сицко А.Л.* |  |  | *Кафедра ИСиТ,  гр. 181071* | | | | |
| *Н.контр.* | | *Шелягович А.С.* |  |  |
| *Утв.* | |  |  |  |