

ВВЕДЕНИЕ

В настоящее время на предприятиях существует потребность в учете рабочего времени персонала в рабочих помещениях на предприятии для контроля рабочего времени и передвижения сотрудников и не идентифицированных лиц.

Системы учета рабочего времени позволяют проанализировать занятость персонала при выполнении рабочих функций в течение рабочего времени, результаты анализа помогают руководству предприятия оптимизировать некоторые рабочие процессы, что повышает производительность труда сотрудников. Контроль передвижений персонала позволяет проанализировать внутреннюю логистику, что приводит к сокращению времени на перемещение как продукции в процессе изготовления на территории предприятия, так и самого персонала, дополнительно система позволит контролировать доступ персонала в различные помещения объекты предприятия.

Целью дипломного проекта является разработка автоматической системы учёта персонала в помещении с использованием технологии распознавания лиц.

Для достижения поставленной цели дипломного проекта необходимо решить следующие задачи:

- изучить методы распознавания лиц, а также их достоинства и недостатки;
- выбрать язык программирования, библиотеки и базы данных, подходящие для разработки системы;
- выбрать оборудование и программное обеспечение для развёртывания системы;
- разработать программный код, для обработки лиц и веб-интерфейса системы;
- сконфигурировать параметры безопасности системы;
- спроектировать варианты масштабируемости и интеграции системы;
- рассчитать экономические затраты на разработку, а также отпускную цену продажи системы.

Дипломный проект выполнен самостоятельно, проверен в системе «Антиплагиат». Процент оригинальности соответствует норме, установленной кафедрой.

1. ОБЗОР ТЕХНОЛОГИИ РАСПОЗНАВАНИЯ ЛИЦ

1.1 Обзор методов распознавания лиц

Распознавание лиц – практическое применение теории распознавания образов, задача которого состоит в автоматической локализации лица на изображении, а также в идентификации персоны по лицу [1]. Поэтому для начала необходимо понимание теории распознавания образов.

Сама теория распознавания образов — это раздел информатики и смежных дисциплин, развивающий основы и методы классификации и идентификации предметов, явлений, процессов, сигналов, ситуаций и т. п. объектов, которые характеризуются конечным набором некоторых свойств и признаков. Необходимость в таком распознавании возникает в самых разных областях - от военного дела и систем безопасности до оцифровки аналоговых сигналов. Проблема распознавания образов приобрела выдающееся значение в условиях информационных перегрузок, когда человек не справляется с линейно-последовательным пониманием поступающих к нему сообщений, в результате чего его мозг переключается на режим одновременности восприятия и мышления, которому свойственно такое распознавание. Неслучайно, таким образом, проблема распознавания образов оказалась в поле междисциплинарных исследований, в том числе в связи с работой по созданию искусственного интеллекта, а создание технических систем распознавания образов привлекает к себе всё большее внимание [2].

В теории распознавания образов можно выделить два направления:

- изучение способностей к распознаванию, которыми обладают живые существа, объяснение и моделирование их;
- развитие теории и методов построения устройств, предназначенных для решения отдельных задач в прикладных целях.

Распознавание образов — это отнесение исходных данных к определённому классу с помощью выделения существенных признаков, характеризующих эти данные, из общей массы данных.

При постановке задач распознавания стараются пользоваться математическим языком, стремясь — в отличие от теории искусственных нейронных сетей, где основой является получение результата путём эксперимента, — заменить эксперимент логическими рассуждениями и математическими доказательствами.

Классическая постановка задачи распознавания образов: дано множество объектов. Относительно них необходимо провести

классификацию. Множество представлено подмножествами, которые называются классами. Заданы: информация о классах, описание всего множества и описание информации об объекте, принадлежность которого к определённому классу неизвестна. Требуется по имеющейся информации о классах и описании объекта установить к какому классу относится этот объект.

Наиболее часто в задачах распознавания образов рассматриваются монохромные изображения, что дает возможность рассматривать изображение как функцию на плоскости. Если рассмотреть точечное множество на плоскости T , где функция $f(x, y)$ каждой точке изображения его характеристику: яркость, прозрачность, оптическую плотность, - то такая функция есть формальная запись изображения.

Множество же всех возможных функций $f(x, y)$ на плоскости T есть модель множества всех изображений X . Вводя понятие сходства между образами, можно поставить задачу распознавания. Конкретный вид такой постановки сильно зависит от последующих этапов при распознавании в соответствии с тем или иным подходом.

Для оптического распознавания образов применяется 3 подхода:

Первый подход - применение метода перебора вида объекта под различными углами, масштабами, смещениями и т. д. Для букв нужно перебирать шрифт, свойства шрифта и т. д.

Второй подход - поиск контура объекта и исследование его свойства (связность, наличие углов и т. д.)

Третий подход - использование искусственные нейронные сети. Этот метод требует либо большого количества примеров задачи распознавания (с правильными ответами), либо специальной структуры нейронной сети, учитывающей специфику данной задачи.

Изучив теорию распознавания образов, становится понятна суть теории распознавания лиц. В то время как люди могут распознавать лица без особых усилий, распознавание лиц представляет собой сложную проблему распознавания образов в вычислительной технике. Системы распознавания лиц пытаются идентифицировать человеческое лицо, которое является трехмерным и меняется в зависимости от освещения и выражения лица, на основе его двумерного изображения. Для выполнения этой вычислительной задачи системы распознавания лиц выполняют четыре этапа. На первом этапе происходит обнаружение первого лица для отделения лица от фона изображения. На втором этапе сегментированное изображение лица выравнивается с учетом позы лица, размера изображения и фотографических

свойств, таких как освещение и оттенки серого. Цель процесса выравнивания — обеспечить точную локализацию черт лица на третьем этапе — извлечении черт лица. Такие элементы, как глаза, нос и рот, определяются и измеряются на изображении для представления лица. Затем установленный таким образом вектор признаков лица на четвертом этапе сопоставляется с базой данных лиц[1].

На рисунке 1 представлен пример того, как выделяется лицо, найденное при помощи библиотеки «OpenCV».



Рисунок 1 - Автоматическое распознавание лиц с помощью библиотеки «OpenCV»

Далее будут приведены методы распознавания лиц:

1. Традиционный. Некоторые алгоритмы распознавания лиц идентифицируют черты лица, извлекая ориентиры или особенности из изображения лица субъекта. Например, алгоритм может анализировать относительное положение, размер и/или форму глаз, носа, скул и челюсти. Эти функции затем используются для поиска других изображений с соответствующими характеристиками. Другие алгоритмы нормализуют галерею изображений лиц, а затем сжимают данные лица, сохраняя на изображении только те данные, которые полезны для распознавания лиц. Затем изображение зонда сравнивается с данными лица. Одна из первых успешных систем основана на методах сопоставления шаблонов, применяемых к набору характерных черт лица, обеспечивая своего рода сжатое представление лица.

Алгоритмы распознавания можно разделить на два основных подхода: геометрический, который рассматривает отличительные признаки, и фотометрический, представляющий собой статистический подход, который преобразует изображение в значения и сравнивает значения с шаблонами для устранения отклонений. Некоторые классифицируют эти алгоритмы на две широкие категории: целостные и основанные на функциях модели. Первые алгоритмы пытаются распознать лицо целиком, в то время как вторые,

основанные на признаках, подразделяются на компоненты. Например, в соответствии с признаками, и анализирует каждый, а также его пространственное положение по отношению к другим признакам.

Популярные алгоритмы распознавания включают в себя анализ основных компонентов с использованием собственных граней, линейный дискриминантный анализ, сопоставление графа упругих пучков с использованием алгоритма «Fisherface», скрытую марковскую модель, многолинейное обучение подпространств с использованием тензорного представления и сопоставление динамических связей, мотивированное нейронами. Современные системы распознавания лиц все чаще используют методы машинного обучения, такие как глубокое обучение. На рисунке 2 изображён пример того, как выделяются собственные грани после обработки изображением.

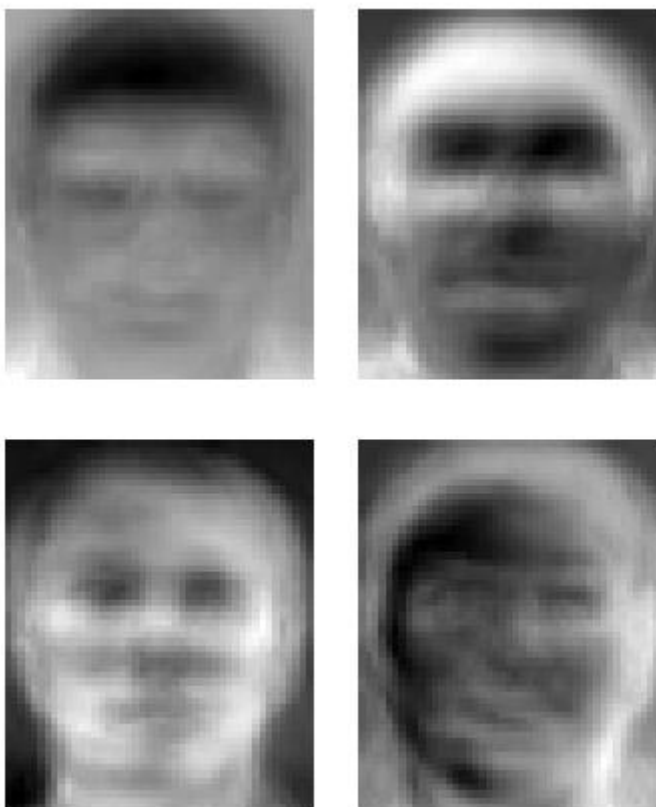


Рисунок 2 – Некоторые собственные грани «AT&T Laboratories»

2. Идентификация человека на расстоянии (HID). Чтобы обеспечить идентификацию человека на расстоянии, изображения лиц с низким разрешением улучшаются с помощью галлюцинаций лица. На снимках видеонаблюдения лица часто очень маленькие. Но поскольку алгоритмы

распознавания лиц, которые идентифицируют и отображают черты лица, требуют изображений с высоким разрешением, были разработаны методы повышения разрешения, позволяющие системам распознавания лиц работать с изображениями, снятыми в условиях с высоким отношением сигнал/шум. Алгоритмы галлюцинаций лица, которые применяются к изображениям до того, как эти изображения будут отправлены в систему распознавания лиц, используют машинное обучение на основе примеров с заменой пикселей или распределением ближайших соседних индексов, которые могут также включать демографические и возрастные характеристики лица. Использование методов галлюцинаций лица улучшает производительность алгоритмов распознавания лиц с высоким разрешением и может использоваться для преодоления ограничений, присущих алгоритмам сверхвысокого разрешения. Техники галлюцинаций лица также используются для предварительной обработки изображений, в которых лица замаскированы. Здесь снимается маскировка, например, солнцезащитные очки, и к изображению применяется алгоритм галлюцинации лица. Такие алгоритмы галлюцинаций лица необходимо обучать на похожих изображениях лиц с маскировкой и без нее. Чтобы заполнить область, обнаруженную при удалении маскировки, алгоритмы галлюцинаций лица должны правильно отображать все состояние лица, что может быть невозможно из-за мгновенного выражения лица, запечатленного на изображении с низким разрешением.

3. Трёхмерное распознавание. Метод трехмерного распознавания лиц использует 3D-сенсоры для сбора информации о форме лица. Эта информация затем используется для определения отличительных черт на поверхности лица, таких как контур глазниц, носа и подбородка. Одним из преимуществ 3D-распознавания лиц является то, что на него не влияют изменения освещения, как на другие методы. Он также может идентифицировать лицо под разными углами обзора, включая вид в профиль. Точки трехмерных данных лица значительно повышают точность распознавания лиц. Исследование трехмерного распознавания лиц стало возможным благодаря разработке сложных датчиков, которые проецируют структурированный свет на лицо. Техника трехмерного сопоставления чувствительна к выражениям, поэтому исследователи из «Техниона» применили инструменты метрической геометрии для обработки выражений как изометрий. В новом методе захвата 3D-изображений лиц используются три камеры слежения, направленные под разными углами; одна камера будет направлена на переднюю часть объекта, вторая сбоку, а третья под углом. Все эти камеры будут работать вместе, чтобы

они могли отслеживать лицо объекта в режиме реального времени и иметь возможность обнаруживать и распознавать лица. На рисунке 3 представлена трёхмерная модель, отсканированного лица.

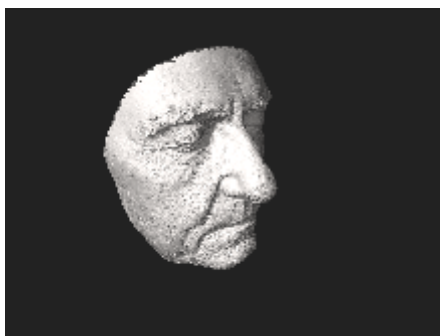


Рисунок 3 - 3D модель человеческого лица

4. Тепловизионные камеры. Другой формой ввода входных данных для распознавания лиц является использование тепловизионных камер. На рисунке 4 можно увидеть то, как отображается лицо человека при съемке тепловизионной камерой. В этой процедуре камеры будут определять только форму головы и игнорировать аксессуары объекта, такие как очки, шляпы или макияж. В отличие от обычных камер, телевизионные камеры могут захватывать изображения лиц даже в условиях низкой освещенности и в ночное время без использования вспышки и раскрытия положения камеры. Однако базы данных для распознавания лиц ограничены. Усилия по созданию баз данных телевизионных изображений лиц были предприняты еще в 2004 г. К 2016 г. существовало несколько баз данных, в том числе «ИИТД-PSE» и база данных термальных лиц Нотр-Дам. Существующие телевизионные системы распознавания лиц не могут надежно обнаружить лицо на телевизионном изображении, сделанном вне помещения [1].



Рисунок 4 - Псевдо цветное изображение двух людей, полученное в длинноволновом инфракрасном (тепловом) свете

В 2018 году исследователи из Исследовательской лаборатории армии США («ARL») разработали метод, который позволяет им сопоставлять изображения лиц, полученные с помощью телевизионной камеры, с изображениями в базах данных, снятыми с помощью обычной камеры. Известный как метод кросс-спектрального синтеза из-за того, как он объединяет распознавание лиц из двух разных модальностей изображения, этот метод синтезирует одно изображение путем анализа нескольких областей лица и деталей. Он состоит из модели нелинейной регрессии, которая отображает определенное тепловое изображение в соответствующее видимое изображение лица, и задачи оптимизации, которая проецирует скрытую проекцию обратно в пространство изображения. Ученые «ARL» отметили, что этот подход работает путем объединения глобальной информации (т. е. особенностей всего лица) с локальной информацией (а именно с особенностью глаз, носа и рта). Согласно тестам производительности, проведенным в лаборатории «ARL», модель кросс-спектрального синтеза для нескольких регионов продемонстрировала улучшение производительности примерно на 30 % по сравнению с базовыми методами и примерно на 5 % по сравнению с современными методами [1].

Вывод: в разрабатываемой системе будет использован метод традиционный метод распознавания лиц. Использование традиционного метода гораздо дешевле других система, т.к. тепловизионные камеры - являются более дорогостоящими, чем обычные, трёхмерное распознавание так же требует дорогостоящих сенсоров. А использование идентификации на расстоянии является не подходящим, из-за ненадёжности алгоритмов распознавания, в сравнении с традиционным методом.

1.2 Обзор систем видеонаблюдения с применением технологий распознавания лиц

В коммерческом секторе системы используются в различных областях. Примеры использования в социальных сетях:

- компания «Lookserу» в 2014 году запустила одноименную программу. Приложение позволяет общаться в видеочате с другими через специальный фильтр для лиц, который изменяет внешний вид пользователей. Приложения для увеличения изображения, уже представленные на рынке, такие как приложения «Facetune» и «Perfect365», были ограничены статическими изображениями, в то время как «Lookserу» позволял использовать дополненную реальность для живых видео. В конце 2015 года «SnapChat»

приобрела «Lookery», которая впоследствии стала отличительной функцией её линз. Приложения-фильтры «Snapchat» используют технологию распознавания лиц, и на основе черт лица, идентифицированных на изображении, на лицо накладывается трехмерная сетчатая маска [1].

- приложение «DeerFace» — это система распознавания лиц с глубоким обучением, созданная исследовательской группой компании «Facebook». Он идентифицирует человеческие лица на цифровых изображениях. Он использует девятислойную нейронную сеть с более чем 120 миллионами весов соединений и был обучен на четырех миллионах изображений, загруженных пользователями социальной сети «Facebook». Точность системы составляет 97% по сравнению с 85% для системы идентификации нового поколения ФБР.

- Алгоритм «TikTok», который одноимённое приложение использовало до 2020 года для распознавания лиц как в видео пользователей, так и для определения возраста, пола и этнической принадлежности.

Так же распознавание лиц связано с использованием услуг проверки личности. Многие компании и другие компании сейчас работают на рынке, чтобы предоставлять эти услуги банкам, ICO и другим электронным предприятиям. Распознавание лиц использовалось как форма биометрической аутентификации для различных вычислительных платформ и устройств. В операционной системе Android 4.0 «Ice Cream Sandwich» добавлено распознавание лиц с помощью передней камеры смартфона в качестве средства разблокировки устройств, в то время как компания «Microsoft» представила вход с распознаванием лиц в свою игровую консоль «Xbox 360» через аксессуар «Kinect», а также операционную систему «Windows 10» через платформу «Windows Hello» (для которой требуется камера с инфракрасной подсветкой). В 2017 году смартфон «Apple iPhone X» представил распознавание лиц в линейке продуктов с помощью платформы «Face ID», в которой используется система инфракрасной подсветки.

Алгоритмы распознавания лиц могут помочь в диагностике некоторых заболеваний по характерным признакам на носу, щеках и других частях лица человека. Опираясь на разработанные наборы данных, машинное обучение использовалось для выявления генетических аномалий только на основе размеров лица. Алгоритм «FRT» также использовался для проверки пациентов перед хирургическими процедурами.

В марте 2022 года, согласно публикации «Forbes», «FDNA», компания-разработчик ИИ, заявила что в течение 10 лет они работали с генетиками над созданием базы данных, содержащей около 5000 заболеваний, 1500 из которых можно обнаружить с помощью алгоритмы распознавания лиц.

Использование системы распознавание лиц широко распространена в государственной сфере в службах безопасности.

Государственный департамент США использует одну из крупнейших в мире систем распознавания лиц с базой данных 117 миллионов взрослых американцев, фотографии которых обычно взяты из фотографий с водительских прав. Хотя проект еще не полностью завершён, его используют в некоторых городах, чтобы дать подсказки относительно того, кто был на фотографии. ФБР использует фотографии в качестве инструмента расследования, а не для достоверной идентификации. По состоянию на 2016 г. распознавание лиц использовалось для идентификации людей на фотографиях, сделанных полицией в Сан-Диего и Лос-Анджелесе (не на видео в реальном времени, а только на фотографиях с бронированием) и планировалось использовать в Западной Вирджинии и Далласе. В последние годы в штате Мэриленд проект использовался для распознавания лиц, сравнивая лица людей с фотографиями на их водительских правах. Эта система вызвала споры, когда ее использовали в Балтиморе для ареста непослушных демонстрантов после смерти Фредди Грея во время содержания под стражей в полиции. Многие другие штаты используют или разрабатывают аналогичную систему, однако в некоторых штатах действуют законы, запрещающие ее использование.

ФБР также запустило свою программу идентификации следующего поколения, включающую распознавание лиц, а также более традиционные биометрические данные, такие как отпечатки пальцев и сканирование радужной оболочки глаза, которые могут извлекаться как из криминальных, так и из гражданских баз данных. Федеральное управление общей подотчетности раскритиковало ФБР за то, что оно не приняло во внимание различные проблемы, связанные с конфиденциальностью и точностью. Начиная с 2018 года Таможенно-пограничная служба США развернула «биометрические сканеры лица» в аэропортах США. Пассажиры, вылетающие международными рейсами, могут пройти регистрацию, проверку безопасности и посадку после того, как будут сняты и проверены изображения лиц путем сопоставления их фотографий на документы, хранящихся в базе данных СВР. Изображения, сделанные для путешественников с гражданством США, будут удалены в течение 12 часов. Управление транспортной безопасности («TSA») выразило намерение принять аналогичную программу для внутренних авиаперевозок в процессе проверки безопасности в будущем. Американский союз гражданских свобод является одной из организаций, выступающих против программы, поскольку программа будет использоваться

в целях наблюдения. В 2019 году исследователи сообщили, что иммиграционная и таможенная служба использует программное обеспечение для распознавания лиц в базах данных государственных водительских прав, в том числе в некоторых штатах, которые предоставляют лицензии иммигрантам без документов. В декабре 2022 года 16 крупных внутренних аэропортов США начали тестировать технологию распознавания лиц, когда киоски с камерами проверяют фотографии на удостоверениях личности путешественников, чтобы убедиться, что пассажиры не являются самозванцами.

1.3 Достоинства и недостатки систем, использующих технологии распознавания лиц

Системы, использующие технологии распознавания лиц, обладают достоинствами, а именно:

1. Алгоритмы, использующиеся в системах распознавания лиц совершенствуются. Так в 2006 году на конкурсе производительности новейших алгоритмов распознавания лиц «Face Recognition Grand Challenge» («FRGC») в тестах использовались изображения лица с высоким разрешением, 3D-сканы лица и изображения радужной оболочки глаза. Результаты показали, что новые алгоритмы в 10 раз точнее, чем алгоритмы распознавания лиц 2002 года, и в 100 раз точнее, чем алгоритмы 1995 года. Некоторые из алгоритмов смогли превзойти участников-людей в распознавании лиц и смогли однозначно идентифицировать однояйцевых близнецов.

2. Для работы системы распознавания лиц не требуется сотрудничество испытуемого, что является одним из ключевых преимуществ системы, позволяя выполнять массовую идентификацию. Правильно спроектированные системы, установленные в аэропортах, мультиплексах и других общественных местах, могут идентифицировать людей среди толпы, причем прохожие даже не подозревают о системе.

Но также системы, использующие технологии распознавания лиц, обладают перечнем недостатков:

1. По сравнению с системами, использующими другие биометрические методы распознавания (сканирование отпечатков пальцев, сканирование сетчатки глаз и т.д.), системы с распознаванием лиц могут быть не самыми надежными и эффективными. Показатели качества очень важны в системах распознавания лиц, поскольку изображения лиц могут сильно различаться. Такие факторы, как освещение, выражение лица, поза и шум во время захвата

лица, могут влиять на работу систем распознавания лиц. Среди всех биометрических систем распознавание лиц имеет самые высокие показатели ложного принятия и отклонения, поэтому были подняты вопросы об эффективности программного обеспечения для распознавания лиц в случаях безопасности на железных дорогах и в аэропортах. Так Ральф Гросс, исследователь из Института робототехники Карнеги-Меллона в 2008 году, описывает одно препятствие, связанное с углом обзора лица: «Распознавание лиц, становится довольно хорошим, если смотреть на лицо полностью и под углом 20 градусов, но как только вы приближаетесь к профилю, были проблемы». Помимо вариаций поз, изображения лиц с низким разрешением также очень трудно распознать. Это одно из основных препятствий для распознавания лиц в системах наблюдения. Распознавание лиц менее эффективно, если выражения лица различаются. Широкая улыбка может сделать систему менее эффективной. Например, в Канаде в 2009 году на фотографиях на паспорт разрешалось только нейтральное выражение лица.

2. Существует также непостоянство в наборах данных, используемых исследователями систем. Исследователи могут использовать от нескольких предметов до десятков предметов и от нескольких сотен изображений до тысяч изображений. Исследователям важно предоставить друг другу наборы данных, которые они использовали, или иметь хотя бы стандартный набор данных.

3. Использование в оценках точности систем выборок меньших размеров, чем это было бы необходимо для крупномасштабных приложений. Поскольку распознавание лиц не совсем точное, создается список потенциальных совпадений. Затем человек-оператор должен просмотреть эти потенциальные совпадения, и исследования показывают, что операторы выбирают правильное совпадение из списка только примерно в половине случаев. Это вызывает проблему нацеливания на неправильного подозреваемого. Однако решается проверкой найденных лиц – человеком, что не вызовет множественные затраты, т.к. число злоумышленников составляет небольшой процент от населения.

Примеры неэффективного использования систем, использующих технологии распознавания лиц, вызванные недостатками данного типа систем:

1. Критики этой технологии жалуются, что схема лондонского района Ньюхэм по состоянию на 2004 год так и не распознала ни одного преступника несмотря на то, что несколько преступников в базе данных системы проживают в этом районе, а система работает уже несколько лет. Насколько известно полиции, автоматическая система распознавания лиц Ньюхэма ни

разу не засекла живую цель. Эта информация кажется противоречащей утверждениям о том, что система позволила снизить уровень преступности на 34% (поэтому она была развернута и в Бирмингеме).

2. Эксперимент, проведенный в 2002 году местным отделением полиции в городе Тампе, штат Флорида, дал такие же неутешительные результаты. Система в бостонском аэропорту Логан была отключена в 2003 году после того, как в течение двухлетнего тестового периода не удалось найти совпадений.

1.4 Разработка диаграммы вариантов использования

Целью разрабатываемой системы является учёт персонала, посещающего и покидающего помещения, с предоставлением веб-интерфейса для просмотра данных учёта.

Разрабатываемая система учёта решает следующий спектр задач:

- опознание лиц, посещающих и покидающих помещение, в соответствии с известными лицами;
- фиксация даты и времени опознания лиц;
- хранение биометрических данных персонала в базе данных;
- обеспечение простоты масштабируемости системы, путём использования метода контейнеризации программных элементов системы;
- предоставление веб-интерфейса для доступа к данным учёта системы;
- разграничение доступа для доступа к данным системы через веб-интерфейс.

На рисунке 5 и листе графического материала ГУИР.466152.001 ПД изображена диаграмма вариантов использования системы, которая была разработана на основании задач и целей разрабатываемой системы, а так примеров схожих систем.

На диаграмме можно увидеть, что функционал, предоставляемый рядовым пользователям системы, заключается лишь в отображении информации, а системным администраторам системы предоставляется полный доступ к базе данных системы.

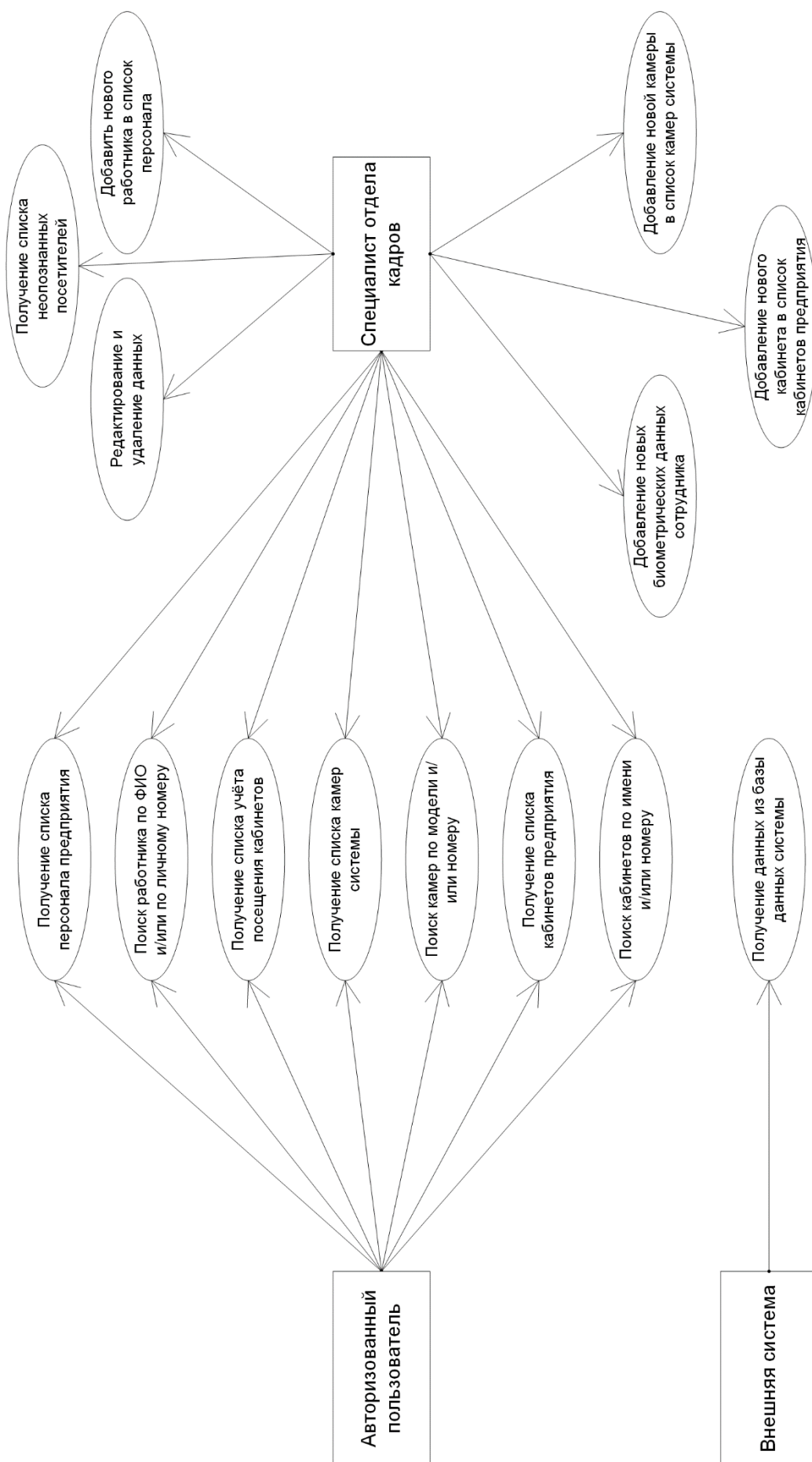


Рисунок 5 – Диаграмма вариантов использования

Вывод: Технологии распознавания лиц является разделом теории распознавания образов. Существует 4 метода распознавания лиц: традиционный (использование алгоритмов, оценивающих особенности лицевого строения людей), идентификация человека на расстоянии (преобразование зашумленных изображений в более качественные их копии), трехмерное (использование трёхмерного сканирования лица), тепловизионные камеры (использование различия температур в разных участках тела и лица человека). Методы распознавания лиц обладают преимуществами: постоянное совершенствование алгоритмов распознавания, отсутствие необходимости согласия людей, чьи лица будут обрабатываться. Однако методы обладают и существенными недостатками: ненадёжность распознавания лиц из-за зависимости изображений от параметров окружающей среды (плотность воздуха, угол падения солнечных лучей, угол расположения камеры к лицу человека и т.п.), непостоянство набора данных (изменение во времени данных, отвечающих за особенности строения лиц и т.п.), ошибочная оценка эффективности метода из-за недостаточной или некачественной выборки исходных данных. Результат работы системы можно посмотреть в веб-интерфейсе, который разделён на пользовательский и административный уровень. Проанализированы требования к разрабатываемой системе и была разработана диаграмма вариантов использования.

2. ВЫБОР ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И ОБОРУДОВАНИЯ ДЛЯ РАЗРАБОТКИ СИСТЕМЫ

2.1 Выбор языка программирования, базы данных и библиотек

Выбор языка программирования и выбор библиотек для разработки приложений тесно взаимосвязан.

Существует 3 основных варианта выбора:

- вначале выбираем язык программирования, исходя из требования по разработке (например, высокая производительность или высокая читаемость программного кода языка), а уже потом выбираются библиотеки, необходимые для разработки. В таком случае, мы можем выбрать язык программирования, удовлетворяющий непобедимым техническим требованиям, но на стадии выбора библиотек может оказаться, что под конкретные задачи библиотеки отсутствуют на выбранном языке, поэтому надо разрабатывать нужные механизмы самим, что сильно влияет на время разработки;

- по заданному списку решаемых задач подбираем необходимые библиотеки, написанные под один язык программирования. Однако данный вариант может привести к тому, что разработка проекта на этом языке будет более затратной по времени разработки, чем выбрать другой язык, с менее обширным объемом библиотек. Также проект, реализованный на этом языке, может оказаться более затратным по техническим ресурсам.

- комплексный анализ библиотек и языков; позволяет подобрать оптимальное сочетание языка программирования и библиотек, для приемлемого решения задач, требующих решения в разработке проекта.

Выбор языка программирования, а также библиотек будет исходить из задач, требующих решения для разработки:

- для захвата видеопотока (желательно захват видеопотока по протоколу «RTSP»);

- поиск объектов(лиц) на изображении;

- анализ принадлежности человеку;

- коннектор к базе данных, хранящей список допущенного персонала с их изображения лиц и время ухода и прихода персонала.

Также стоящие перед нами задачи будут требовать высокой производительности от приложения.

Для осуществления захвата экрана и поиска объектов существуют следующие библиотеки:

- библиотека «OpenCV» - разработанный под языки программирования: Python, C++, Java;

- библиотека «SimpleCV» - разработанный для языка Python;

- библиотека «Accord.NET Framework» - разработанный для языка C#.

Для анализа лиц существуют:

- библиотека «dlib» - разработанная для языка программирования C++;

- библиотека «face_recognition» - разработанная для языка программирования Python;

- библиотека «libfacedetection» - разработанный для языка Python, C++.

Выбор коннекторов к базам данных является несложной задачей, т.к. почти все популярные базы данных имеют коннекторы к множеству языков программирования.

Для достижения максимальной производительности приложения, исходя из представленных выше библиотека и поддерживаемых ими языков, а также, учитывая представленный график (см. рисунок 6) подойдёт C++. Однако, имея проблемы с кроссплатформенностью, а именно, что под каждый тип операционной системы нужно отдельно компилировать программу, то данный язык программирования не подходит. Т.к. у перечисленных библиотек есть другой общий язык программирования – Python, то на нём и будет разрабатываться проект. А библиотеки будут выбраны следующие: «OpenCV», «face_recognition».

Преимущества языка Python [3]:

- динамическая типизация;
- автоматическое управление памятью;
- обширная база данных библиотек;
- минималистичный синтаксис ядра;
- высокая читаемость кода;
- мультиплатформенность программного кода;
- полностью объектно-ориентированный;
- наличие удобного загрузчика библиотек – pip.

Недостатки языка Python:

- более низкая скорость работы, в сравнении с компилируемыми языками;
- более высокое потребление оперативной памяти, в сравнении с компилируемыми языками.

Программная часть проекта будет использовать многопоточность – это свойство платформы (например, операционной системы, виртуальной машины и т. д.) или приложения, состоящее в том, что процесс, порождённый в операционной системе, может состоять из нескольких потоков,

выполняющихся «параллельно», то есть без предписанного порядка во времени. При выполнении некоторых задач такое разделение может достичь более эффективного использования ресурсов вычислительной машины [4].

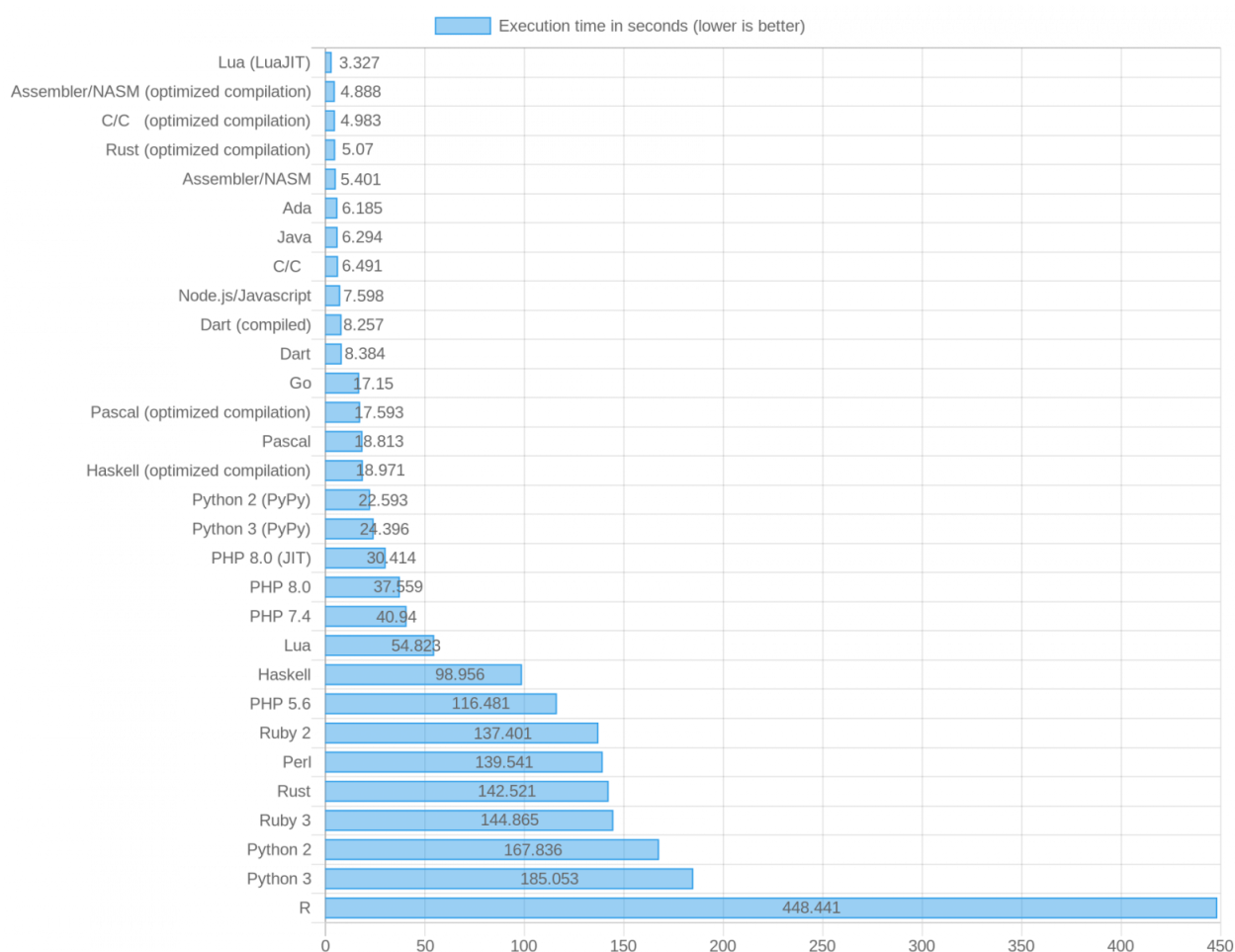


Рисунок 6 – График скорости выполнения теста на основе перебора простых чисел для различных языков программирования

Сутью многопоточности является квази многозадачность на уровне одного исполняемого процесса, то есть все потоки выполняются в адресном пространстве процесса. Кроме этого, все потоки процесса имеют не только общее адресное пространство, но и общие дескрипторы файлов. Выполняющийся процесс имеет как минимум один (главный) поток.

Многопоточность (как доктрину программирования) не следует путать ни с многозадачностью, ни с многопроцессорностью, несмотря на то, что операционные системы, реализующие многозадачность, как правило, реализуют и многопоточность.

К достоинствам многопоточной реализации той или иной системы перед многозадачной можно отнести следующее:

- упрощение программы в некоторых случаях за счёт использования общего адресного пространства;
- меньшие относительно процесса временные затраты на создание потока.

К достоинствам многопоточной реализации той или иной системы перед однопоточной можно отнести следующее:

- упрощение программы в некоторых случаях, за счёт вынесения механизмов чередования выполнения различных слабо взаимосвязанных подзадач, требующих одновременного выполнения, в отдельную подсистему многопоточности [5];
- повышение производительности процесса за счёт распараллеливания процессорных вычислений и операций ввода-вывода [5].

В случае, если потоки выполнения требуют относительно сложного взаимодействия друг с другом, возможно проявление проблем многозадачности, таких как взаимные блокировки.

Библиотека «OpenCV» - это библиотека алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом [6]. Может свободно использоваться в академических и коммерческих целях — распространяется в условиях лицензии «BSD». Области применения библиотеки «OpenCV» включает:

- наборы инструментов для 2D- и 3D-функций
- оценка одометрии;
- система распознавания лиц;
- распознавание жестов;
- взаимодействие человека и компьютера (HCI);
- мобильная робототехника;
- понимание движения;
- обнаружение объекта;
- сегментация и распознавание;
- стереозрение «Stereopsis»: восприятие глубины с 2-х камер;
- структура от движения (SFM);
- отслеживание движения;
- дополненная реальность.

Библиотека «Face_recognition» - библиотека, предназначенная для распознавания лиц на изображениях [7]. Основана на базе библиотеки «dlib», созданная с помощью глубокого обучения. Модель обладает точность 99.38% в тесте «Labeled Faces in the Wild benchmark». Библиотека обладает следующими возможностями:

- поиск лиц на изображениях;
- поиск и манипуляции черт лиц на изображения;
- идентификация лиц на изображениях.

2.2 Выбор программного обеспечения и оборудования для развёртывания системы

Для размещения системы будет использоваться технология контейнеризации. Контейнеризация - метод виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного. Эти экземпляры (обычно называемые контейнерами или зонами) с точки зрения выполняемых в них процессов идентичны отдельному экземпляру операционной системы.

Преимущества контейнеризации:

- обеспечении ядром системы полной изолированности контейнеров, поэтому программы из разных контейнеров не могут воздействовать друг на друга [8], на рисунке 7 схематично показана структура реализации приложения Docker на Linux;
- отсутствие дополнительных ресурсных накладных расходов на эмуляцию виртуального оборудования и запуск полноценного экземпляра операционной системы, характерных при аппаратной виртуализации [8].

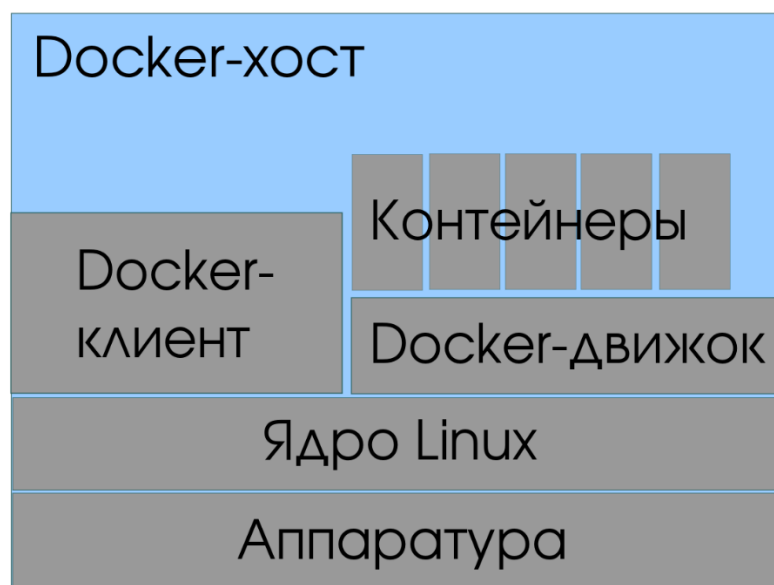


Рисунок 7 – Docker на физическом Linux-сервере

Главным недостатком контейнеризации в отличие от аппаратной виртуализации, при которой эмулируется аппаратное окружение и может быть запущен широкий спектр гостевых операционных систем, выражен в том, что в контейнере может быть запущен экземпляр операционной системы только с тем же ядром, что и у операционной системы на хостинговой машине (все контейнеры узла используют общее ядро).

Существуют реализации, ориентированные на создание практически полноценных экземпляров операционных систем («Solaris Containers», контейнеры «Virtuozzo», «OpenVZ»), так и варианты, фокусирующиеся на изоляции отдельных сервисов с минимальным операционным окружением (jail, Docker). В таблице 1 приведён список программ, реализующих контейнеризацию на различных операционных системах.

Таблица 1 – Список программ, предназначенных для работы с контейнерами

Механизм	Операционная система	Дата выпуска	Лицензия
1	2	3	4
chroot	Unix-системы	1982	Аналогичная ОС
Docker	Linux, FreeBSD, Windows, macOS	2013	Apache 2.0
Solaris Containers	Solaris, OpenSolaris	2005	CDDL
iCore Virtual Accounts	Windows XP	2008	Проприетарное
LXC	Linux	2008	GNU GPL v.2.0
OpenVZ	Linux	2005	GNU GPL v.2.0
FreeBSD Jail	FreeBSD	2000	BSD
WPAR	AIX	2007	Проприетарное

В связи с опытом и простотой использования выбрано приложение Docker. Это программное обеспечение для автоматизации развёртывания и управления приложениями в средах с поддержкой контейнеризации - контейнеризатор, который позволяет «упаковать» приложение со всем его окружением и зависимостями в контейнер, который может быть развёрнут на любой Linux-системе с поддержкой контрольных групп в ядре, а также предоставляет набор команд для управления этими контейнерами [9].

Так же для развёртывания системы будет использоваться такой инструмент как Docker-Compose. Это инструмент для определения и запуска много контейнерных приложений в Docker. Он использует файлы YAML для

настройки служб приложения и выполняет процесс создания и запуска всех контейнеров с помощью одной команды. Утилита CLI позволяет пользователям запускать команды для нескольких контейнеров одновременно, например, создавать образы, масштабировать контейнеры, запускать остановленные контейнеры и т. д. Команды, связанные с манипулированием изображениями или параметрами взаимодействия с пользователем, не имеют значения в приложении Docker-Compose, поскольку они относятся к одному контейнеру. Файл `docker-compose.yml` используется для определения служб приложения и включает в себя различные параметры конфигурации. Например, «build» параметр определяет параметры конфигурации, такие как путь к файлу `Dockerfile`, «command» параметр позволяет переопределять команды программы Docker по умолчанию и многое другое.

Для написания кода будет использоваться программа «Pycharm». Это кроссплатформенная интегрированная среда разработки для языка программирования Python, разработанная компанией «JetBrains» на основе «IntelliJ IDEA» [10]. Предоставляет пользователю комплекс средств для графического отладчика и работы с кодом. Возможности:

- помощь в кодировании и анализ, с завершением кода, подсветкой синтаксиса и ошибок, интеграцией линтера и быстрыми исправлениями;
- навигация по проекту и коду: специализированные представления проекта, представления файловой структуры и быстрый переход между файлами, классами, методами и использованиями;
- поддержка веб-фреймворков: «Django», «web2py» и «Flask»
- интегрированное модульное тестирование с построчным покрытием
- отладка кода при помощи отладчика «PyDev»;
- рефакторинг кода: включая переименование, извлечение метода, введение переменной, введение константы, подтягивание вверх, нажатие вниз и другие;
- поддержка систем «Git», «SVN», «Mercurial»;
- авто дополнение кода.

В качестве системы управления пакетами будет использоваться программа «pip» («Package Installer for Python»), осуществляющий функции поиска, скачивания и обновления библиотек.

Операционной системой для развёртывания контейнеров будет использоваться операционная система «Ubuntu» версии 20.04. Она основана на системе «Debian». Ориентирована на удобство и простоту использования. Она включает широко распространённое использование утилиты «sudo»,

которая позволяет пользователям выполнять администраторские задачи, не запуская потенциально опасную сессию суперпользователя [11].

Преимущества операционной системы «Ubuntu»:

- безопасность;
- наличие обширной базы решений различных проблем, возникающих при эксплуатации системы;
- низкие системные требования;
- постоянная поддержка со стороны разработчиков;
- бесплатная модель распространения.

Базой данных, для хранения списка пользователей, их лицевой метрики, будет использоваться PostgreSQL. Это бесплатная система управления реляционными базами данных с открытым исходным кодом (СУБД) с упором на расширяемость и соответствие SQL. Программа библиотека PostgreSQL библиотека поддерживает транзакции со свойствами «Atomity», «Consistency», «Isolation», «Durability» («ACID»), автоматически обновляемыми представлениями, материализованными представлениями, триггерами, внешними ключами и хранимыми процедурами [12]. Он предназначен для обработки различных рабочих нагрузок, от отдельных машин до хранилищ данных или веб-служб с множеством одновременных пользователей. Программа PostgreSQL является обладает более быстрой скоростью обработки таблиц нежели программа MySQL, исходя из тестов. Далее приведены графики сравнения времени выполнения различных операций в СУБД MySQL и PostgreSQL.

В случае добавления записей PostgreSQL оказался быстрее в среднем в 3,46 раза (рисунок 8). При внутреннем объединении на небольшом количестве строк СУБД PostgreSQL отрывается от СУБД MySQL незначительно (в районе 25 процентов), но с увеличением размера второй таблицы преимущество СУБД PostgreSQL становится более очевидным. При достижении второй таблицей максимального размера скорость объединения в СУБД PostgreSQL уже в 2.8 раз выше. В среднем СУБД PostgreSQL оказалась быстрее в 1,66 раза (рисунок9) [13].

Для получения информации(кадров) с камер будет использоваться протокол RTSP. Являющийся прикладным протоколом, предназначенным для мультимплексирования и пакетирования мультимедийных транспортных потоков (таких как интерактивные медиа, видео и аудио) по подходящему транспортному протоколу и использования в системах, работающих с мультимедийными данными (мультимедийным содержимым, медиасодержимым), и позволяющий удалённо управлять потоком данных с

сервера, предоставляя возможность выполнения команд, таких как запуск (старт), приостановку (пауза) и остановку (стоп) вещания (проигрывания) мультимедийного содержимого, а также доступа по времени к файлам, расположенным на сервере.

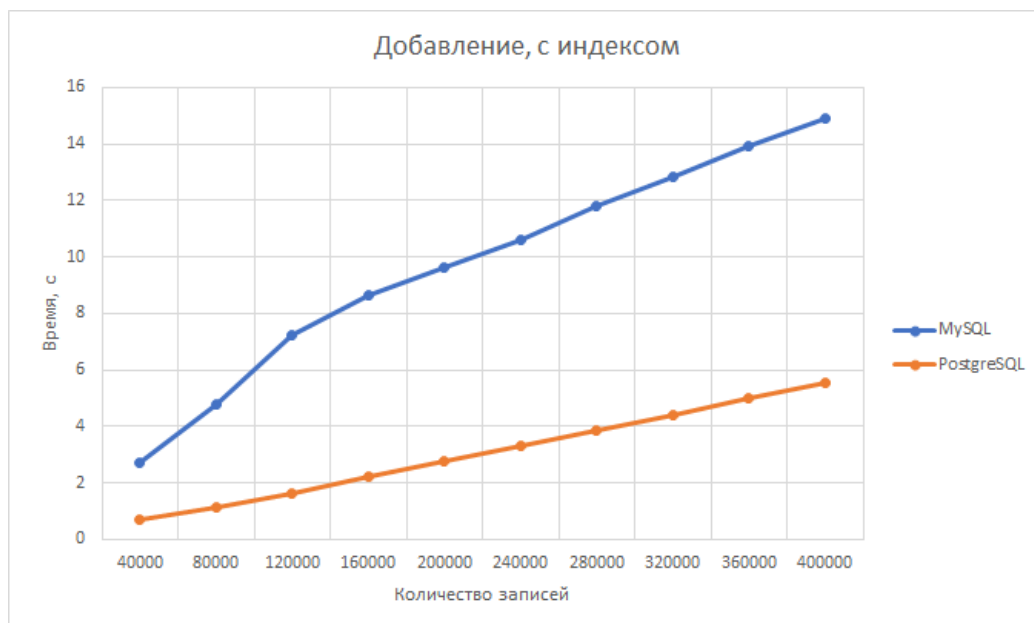


Рисунок 8 – График производительности, при выполнении добавлении записей в таблицу.

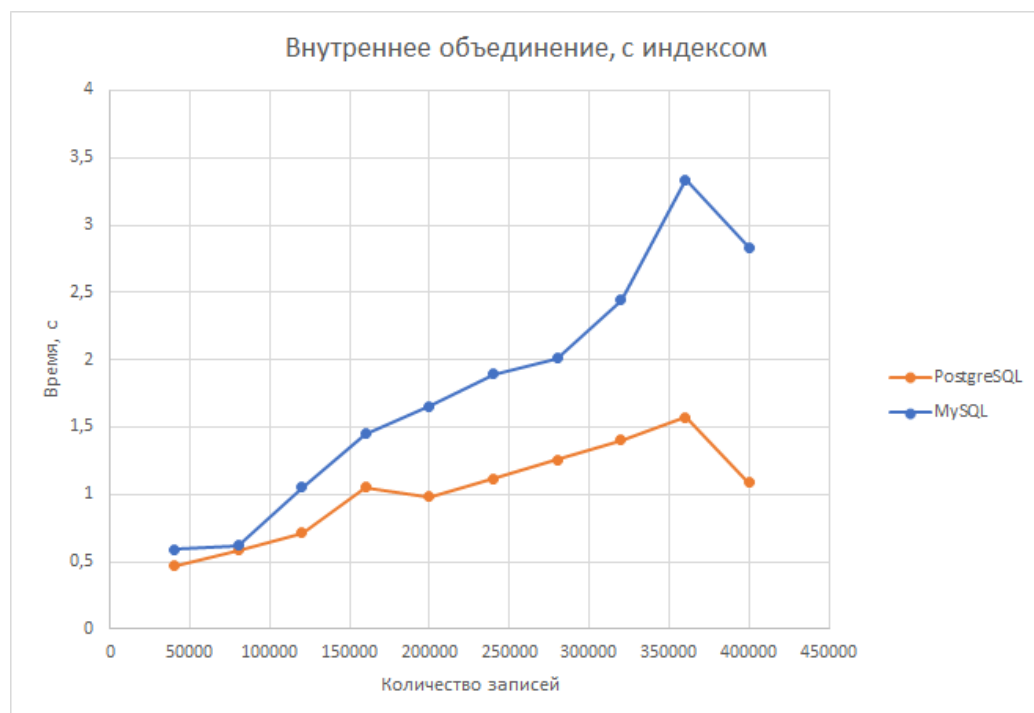


Рисунок 9 – График производительности, при внутреннем объединении записей

Сама по себе передача потоковых данных не является задачей RTSP. Большинство серверов RTSP используют транспортный протокол реального времени (RTP) в сочетании с протоколом управления в реальном времени (RTCP) для доставки медиапотока [14].

Веб-интерфейс разрабатывался с помощью «Django», являющийся свободным фреймворком для веб-приложений на языке Python, использующий шаблон проектирования MVC [15]. На рисунке 10 представлена схема работы модулей системы MVC.

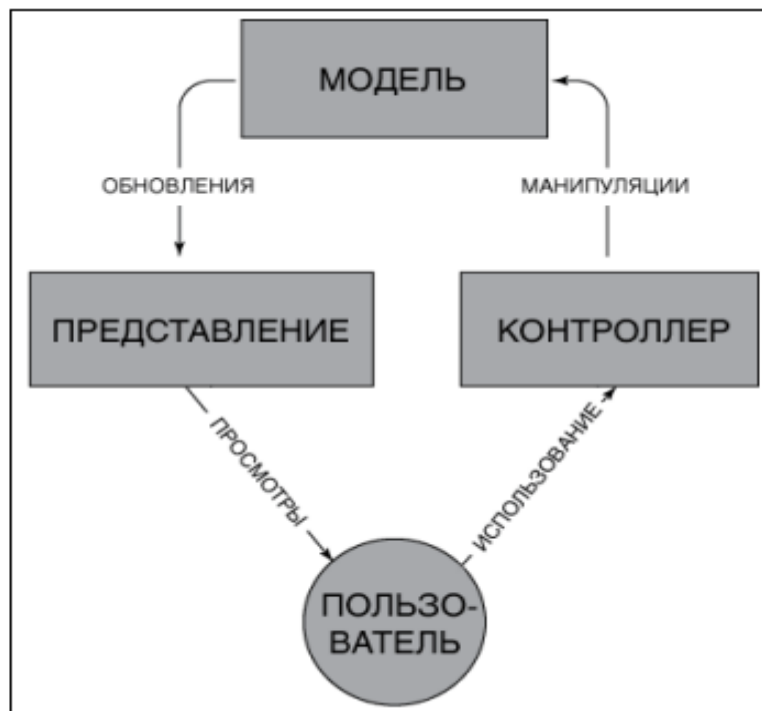


Рисунок 10 – Схема Model-View-Controller

Это схема разделения данных приложения и управляющей логики на три отдельных компонента, таким образом, что модификация каждого компонента может осуществляться независимо:

- модель (Model) предоставляет данные и реагирует на команды контроллера, изменяя своё состояние;
- представление (View) отвечает за отображение данных модели пользователю, реагируя на изменения модели;
- контроллер (Controller) интерпретирует действия пользователя, оповещая модель о необходимости изменений.

Сайт на «Django» строится из одного или нескольких приложений, которые рекомендуется делать отчуждаемыми и подключаемыми. Это одно из существенных архитектурных отличий этого фреймворка от некоторых

других. Один из основных принципов фреймворка — DRY (англ. «Don't repeat yourself»). Также, в отличие от других фреймворков, обработчики URL в «Django» конфигурируются явно при помощи регулярных выражений. Для работы с базой данных «Django» использует собственный ORM, в котором модель данных описывается классами языка Python, и по ней генерируется схема базы данных.

Возможности:

- доступа API к БД с поддержкой транзакций;
- встроенный интерфейс администратора, с уже имеющимися переводами на многие языки;
- диспетчер URL на основе регулярных выражений;
- расширяемая система шаблонов с тегами и наследованием;
- система кеширования;
- интернационализация;
- подключаемая архитектура приложений, которые можно устанавливать на любые Django-сайты;
- шаблоны функций контроллеров «generic views»;
- авторизация и аутентификация, подключение внешних модулей аутентификации: «LDAP», «OpenID»;
- система фильтров («middleware») для построения дополнительных обработчиков запросов, как например включённые в дистрибутив фильтры для кеширования, сжатия, нормализации URL и поддержки анонимных сессий
- библиотека для работы с формами (наследование, построение форм по существующей модели БД);
- встроенная автоматическая документация по тегам шаблонов и моделям данных, доступная через административное приложение.

За отображение веб-интерфейса будет отвечать сочетание языка гипертекстовой разметки «HTML» и каскадных таблиц стилей «CSS».

Преимущества сочетания:

- структуризация элементов веб-интерфейса [16];
- разграничение представления информации от её расположения [17];
- поддержка любого веб-браузера;
- высокий уровень безопасности данных;
- высокая стабильность;
- кэширование стиля страницы, для повышения скорости загрузки страницы;

- централизованное хранение данных – в одном .html файле описывается расположение всех элементов страницы, а в одном .css файле хранится описание вида всех элементов страницы.

Фреймворк Django для работы с базой данных использует программу Redis. Это резидентная система управления базами данных (размещает базу данных в оперативной памяти) класса NoSQL с открытым исходным кодом, работающая со структурами данных типа «ключ — значение». Используется как для баз данных, так и для реализации кэшей, брокеров сообщений [18].

Ориентирована на достижение максимальной производительности на атомарных операциях (заявляется о приблизительно 100 тыс. SET- и GET-запросов в секунду на Linux-сервере начального уровня). Написана на языке C, интерфейсы доступа созданы для большинства основных языков программирования.

Хранит базу данных в оперативной памяти, снабжена механизмами снимков и журналирования для обеспечения постоянного хранения (на дисках, твердотельных накопителях). Также предоставляет операции для реализации механизма обмена сообщениями в шаблоне «издатель-подписчик»: с его помощью приложения могут создавать каналы, подписываться на них и помещать в каналы сообщения, которые будут получены всеми подписчиками (как IRC-чат). Поддерживает репликацию данных с основных узлов на несколько подчинённых. Также поддерживает транзакции и пакетную обработку команд (выполнение пакета команд, получение пакета результатов).

Работает на большинстве POSIX-систем, таких как «Linux», «BSD», «Mac OS X» без каких-либо дополнений, компания-спонсор проекта поддерживает систему на «Linux» и «Mac OS X». Официальной поддержки для сборок «Windows» нет, но доступны некоторые опции, позволяющие обеспечить работу Redis на этой системе, сообщается о работах Microsoft по переносу Redis на «Windows».

В версии 2.6.0 добавлена поддержка скриптового языка Lua, позволяющего выполнять запросы на сервере. Язык Lua позволяет атомарно совершить произвольную обработку данных на сервере и предназначена для использования в случае, когда нельзя достичь того же результата с использованием стандартных команд. Среди языков программирования, имеющих библиотеки для работы с Redis - Си, C++, C#, Clojure, Лисп, Erlang, Java, JavaScript, Haskell, Lua, Perl, PHP, Python, Ruby, Scala, Go, Tcl, Rust, Swift, Nim.

Все данные программы Redis хранит в виде словаря, в котором ключи связаны со своими значениями. Одно из ключевых отличий программы Redis от других хранилищ данных заключается в том, что значения этих ключей не ограничиваются строками. Поддерживаются следующие абстрактные типы данных: строки, списки, множества, хеш-таблицы, упорядоченные множества.

Тип данных определяет, какие операции (команды) доступны для него. Так же поддерживаются такие высокоуровневые операции, как объединение и разность наборов, сортировка наборов.

Для обеспечения стабильности соединения и целостной работы с запросами используется веб-сервер «Nginx». Это веб-сервер и почтовый прокси-сервер, работающий на Unix-подобных операционных системах (тестировалась сборка и работа на «FreeBSD», «OpenBSD», «Linux», «Solaris», «macOS», «AIX» и «HP-UX») [19].

Основные функции веб-сервера «Nginx», при использовании как HTTP сервера:

- обслуживание неизменяемых запросов, индексных файлов, автоматическое создание списка файлов, кэш дескрипторов открытых файлов;
- акселерированное проксирование без кэширования, простое распределение нагрузки и отказоустойчивость;
- поддержка кеширования при акселерированном проксировании и FastCGI;
- акселерированная поддержка FastCGI и memcached-серверов, простое распределение нагрузки и отказоустойчивость;
- модульность, фильтры, в том числе сжатие (gzip), byte-ranges (докачка), chunked-ответы, HTTP-аутентификация, SSI-фильтр;
- несколько подзапросов на одной странице, обрабатываемых в SSI-фильтре через прокси или FastCGI, выполняются параллельно;
- поддержка SSL;
- поддержка PSGI, WSGI;
- экспериментальная поддержка встроенного Perl.

В веб-сервере «Nginx» рабочие процессы обслуживают одновременно множество соединений, мультиплексируя их вызовами операционной системы select, epoll (Linux) и kqueue (FreeBSD). Рабочие процессы выполняют цикл обработки событий от дескрипторов. Полученные от клиента данные разбираются с помощью конечного автомата. Разобранный запрос последовательно обрабатывается цепочкой модулей, задаваемой конфигурацией. Ответ клиенту формируется в буферах, которые хранят данные либо в памяти, либо указывают на отрезок файла. Буфера

объединяются в цепочки, определяющие последовательность, в которой данные будут переданы клиенту.

Все вышеописанные особенности позволяют создавать относительно несложные сайты на языке Python не изучая php или другие языки программирования, заточенные под создание backend разделов веб-приложений.

Разрабатываемая система состоит из оборудования следующих типов:

- сетевое - для организации компьютерной сети между компонентами системы;
- серверное - для обработки данных с камер и предоставления веб-интерфейса и базы данных;
- камеры - для видеофиксации лиц персонала.

Для более подробного понимания необходимого оборудования была разработана структурная схема системы, представленная на рисунке 11 и листе графического материала ГУИР.466152.003 ПД. Диаграмма показывает устройство компьютерной сети разрабатываемой системы. В систему введены маршрутизаторы для создания отдельных подсетей, для обеспечения безопасности и независимости групп элементов сети.

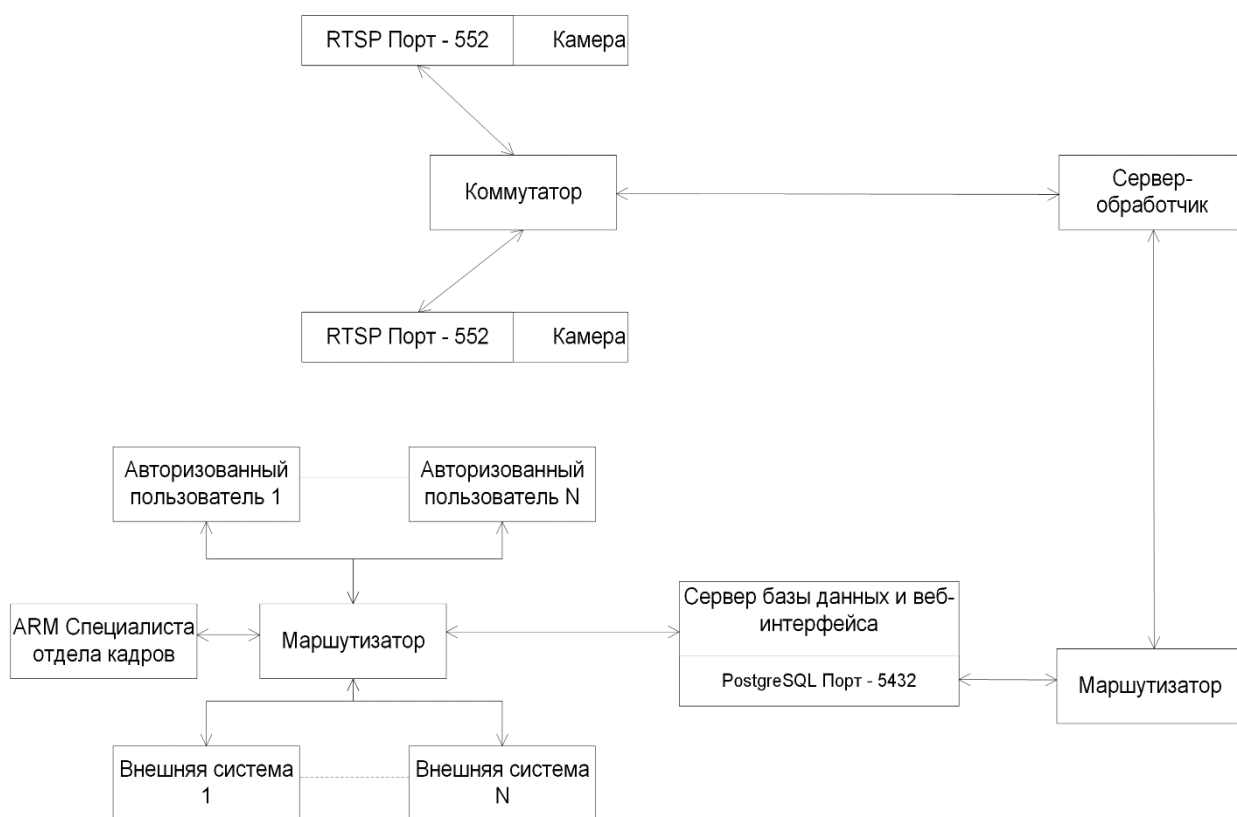


Рисунок 11 – Структурная схема системы

Исходя диаграммы список оборудования будет состоят из следующего оборудования: двух камеры, одного коммутатора, одного сервера-обработчика, одного сервера базы данных.

Камеры должны соответствовать нескольким условиям:

- поддержка RTSP протокола;
- минимальное разрешение передаваемого видеопотока – 640 на 480 пикселей;
- минимальная частота передаваемого видеопотока – 10 кадров в секунду, с возможностью её регулировки;
- наличие Web-интерфейса, по которому можно производить настройку камеры;
- применение внутри помещения – т.к. офисное помещение расположено внутри здания;
- проводной сетевой интерфейс – для более стабильной передачи данных и высоко уровня безопасности, в сравнении с беспроводным интерфейсом;
- стандартная конструкция – для монтажа на уровне средней высоты лица сотрудников;
- наличие ИК-подсветки – для работы в условиях отключения освещения;
- формат сжатия видео «H.264» - для обеспечения приемлемого качества получаемого изображения.

Исходя из вышеописанных требований выбрана модель камеры – «Hikvision DS-2CD2046G2-IU/SL(C)».

Коммутатор, необходимый для соединения камер и маршрутизатора в единый сегмент сети, должен быть настраиваемым либо управляемым, для настройки работающих портов подключения. Исходя из требования выбрана модель – «TP-Link TL-SG105E v3».

Для маршрутизаторов, которые предназначены для организации подсетей внутри всей компьютерной сети системы, необходимо чтобы не было возможности поддержки портами подключения одновременно подключений «WAN» и «LAN» типов, чтобы исключить человеческий фактор при монтаже системы. Исходя из требования был выбран маршрутизатор – «Mikrotik Hex Lite (RB750r2)».

Сервер-обработчик должен обладать следующими требованиями:

- 8 ядер центрального процессора – для работы двух контейнеров, обрабатывающих данные с камер;
- наличие оперативной памяти типа «DDR4» и объёмом 8 Гб – для обеспечения скорости работы контейнеров;

- наличие твердотельного накопителя объёмом 125 Гб – для быстроты работы операционной системы.

Исходя из вышеописанных требований была выбрана модель – «Jet Office 7i10700D8SD12VGALW50».

Для сервера базы данных и веб-интерфейса применяются аналогичные требования, что и к серверу-обработчику, но добавляется наличие жёсткого диска, для расположения на нём базы данных. Для чего была выбрана модель «Jet Office 7i10700D8HD05SD12VGALW50».

На каждом из серверов устанавливаются программы «Docker», «Docker-compose», с помощью которых разворачиваются контейнеры с программами. На рисунке 12 и листе графического материала ГУИР.466152.005 ПД представлена диаграмма компонентов, на которой показана разрабатываемая система, разбитая на структурные компоненты, также на диаграмме представлены связи между структурными компонентами, показывающие потоки данных, их направление и предназначение. На диаграмме структурные компоненты представлены не только программами, а также и оборудованием, за исключением промежуточного сетевого оборудования – маршрутизаторов и коммутатора.

Вывод: Для разработки системы используется язык программирования Python, использование которого позволяет упростить разработку системы. Для хранения данных учёта система использует базу данных PostgreSQL, обеспечивающей быстроту обработки данных, в сравнении с другими базами данных SQL-типа. Для разработки веб-интерфейса системы используется фреймворк «Django», обеспечивающий удобство разработки интерфейса, веб-страница представлена в виде html-страницы с использованием таблиц стилей css. Для развёртывания системы на операционной системе серверов используется программа Docker, позволяющая быстро и безопасно развёртывать программное обеспечение. Разработаны диаграмма сетевой топологии и диаграмма компонентов. Для развёртывания системы было выбрано следующее оборудование:

- две камеры «Hikvision DS-2CD2046G2-IU/SL(C)»;
- одного коммутатора «TP-Link TL-SG105E v3»;
- два маршрутизатора «Mikrotik Hex Lite (RB750r2)»;
- один сервер-обработчик «Jet Office 7i10700D8SD12VGALW50»;
- один сервер базы данных и веб-интерфейса «Jet Office 7i10700D8HD05SD12VGALW50».

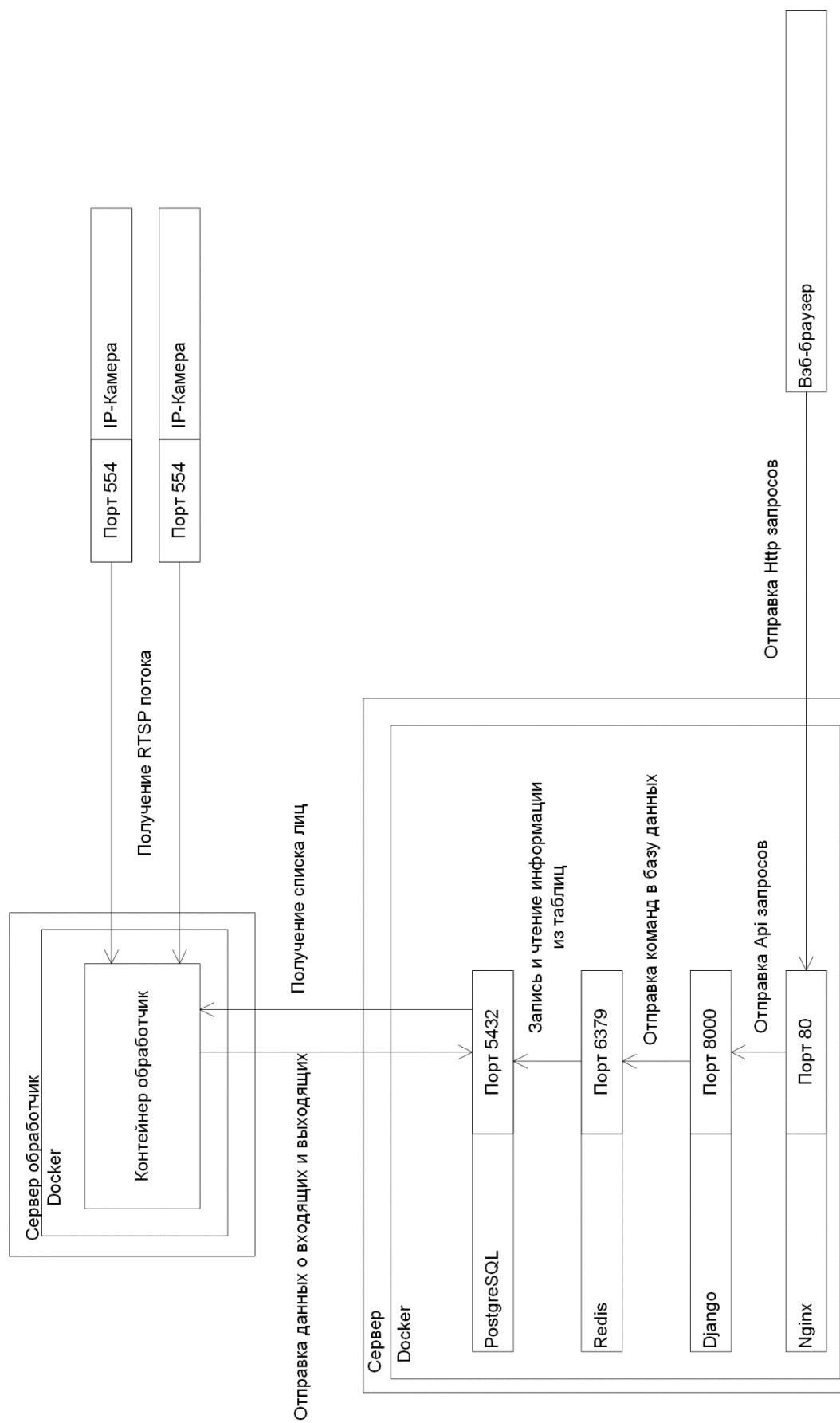


Рисунок 12 – Диаграмма компонентов

3. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ АВТОМАТИЧЕСКОЙ СИСТЕМЫ УЧЁТА ПЕРСОНАЛА

3.1 Структуризация компонентов программного кода системы

Вся программная часть системы разделяется на 2 части, которые работают независимо друг от друга, но взаимодействующих с одной базой данных: веб-интерфейс и обработка данных с камеры.

В свою очередь веб-интерфейс делится на 2 части: frontend и backend.

Backend представлен в виде api-сервера, реализованного с помощью фреймворка «Django». Он позволяет сформировать таблицы в базе данных, путём описания их в виде классов в Python, которые в дальнейшем программа будет использовать как интерфейс доступа к базе данных и на основе которого фреймворк создаст таблицы в базе данных. Фреймворк позволяет обрабатывать api-запросы к базе данных, которые он будет получать с frontend части, далее забирать информацию с базы данных (которая была запрошена в запросе), а потом отправляет её в виде ответа на api-запрос. На схеме диаграммы классов из списка графических документов представлены классы, создаваемые фреймворком. Так же в нём производится настройка администраторского доступа к базе данных.

Frontend представлен в виде 3 файлов:

- файл index.html – представляет собой файл разметки, который отвечает за расположение элементов на странице и их связь со скриптами в файле app.js;
- файл style.css – файл стилей, в котором описан внешний вид элементов страницы;
- файл app.js – файл, хранящий скрипты, с помощью которых страница получает информацию с backend.

Обработка данных с камер является ресурсно-затратным со стороны использования центрального процессора компьютера. Для повышения производительности программы используется метод многопоточной разработки

Первая часть проекта представляет собой алгоритм, состоящий из следующих шагов:

- получение кадра из видеопотока с камеры;
- поиск лиц на кадре;
- анализ найденных лиц на совпадение с персоналом предприятия;
- отправка в базу данных информации о найденных лицах;
- обновление лицевых данных из базы данных.

Для получения кадра используется библиотека «OpenCV», а именно функция VideoCapture, в аргументы которой мы передаём адрес камеры.

Для разделения программы на несколько потоков используется библиотека «Thread», являющейся стандартной библиотекой языка.

Для поиска лица используется библиотека «face_recognition», а именно функция «face location», в которую в качестве аргументов мы отправляем кадр, полученный из «OpenCV» и который перед этим будет сжат, для увеличения работоспособности.

Далее идёт сравнение полученного лица, с лицами, находящимися в буфере. При совпадении, отправляется строка в базу данных, о прохождении персонала у камеры.

Так же в параллельном потоке работает таймер, который раз в 24 часа проверяет внутренний буфер лиц программы на актуальность, а при наличии новых загружает их из базы данных.

3.2 Разработка базы данных

В СУБД PostgreSQL находится база данных «cvddb», которая хранит в себе всю информацию, необходимую на работы система – список сотрудников, их биометрия лиц и т.д. На рисунке 13 и листе графического материала ГУИР.466152.006 ПД представлена диаграмма базы данных. База данных состоит из следующих таблиц:

- таблица «user»;
- таблица «department»;
- таблица «personal»;
- таблица «faces»;
- таблица «cabinets»;
- таблица «cameras»;
- таблица «in_out_date».

Таблица «user» предназначена для хранения всего списка физических лиц, состоит из 5 колонок:

- поле first_name - типа text, хранит имя работника;
- поле last_name - типа text, хранит фамилию работника;
- поле email - типа text, хранит email адрес работника;
- поле age - типа int, хранит возраст работника.

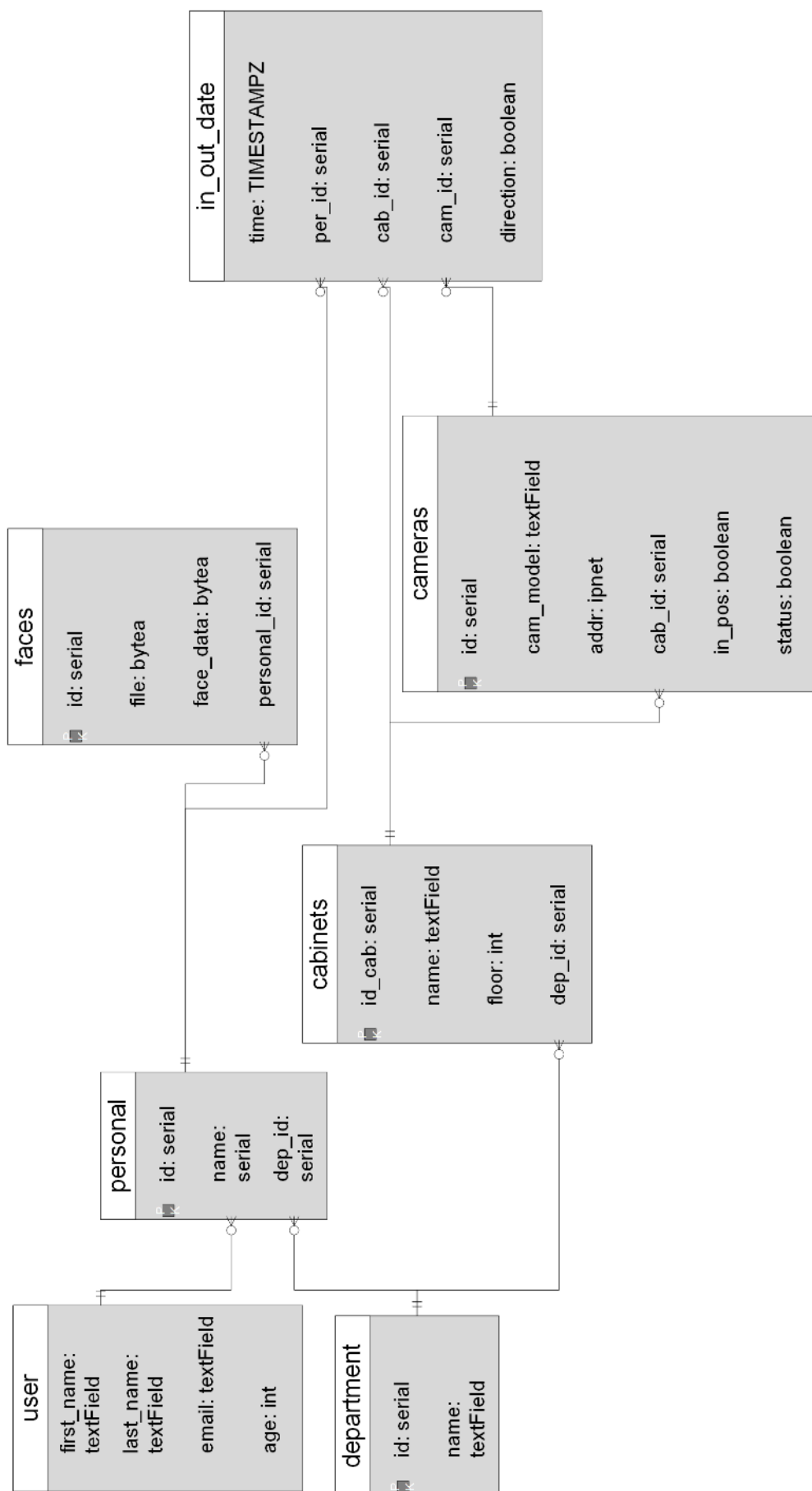


Рисунок 13 – Диаграмма базы данных

Таблица «department» хранит список отделов предприятия, состоит из 2 колонок:

- поле id - типа serial, авто заполняемый уникальный идентификатор отдела, является первичным ключом таблицы;
- поле name – типа text, хранит название отдела.

Таблица «personal» хранит список персонала, который работает в каждом отделе, состоит из 3 колонок:

- поле id - типа serial, авто заполняемый уникальный идентификатором персонала, является первичным ключом таблицы;
- поле name – типа serial, ссылка на имя сотрудника;
- поле dep_id - типа serial, хранит ссылку на отдел, к которому относится сотрудник.

Таблица «faces» хранит лицевые параметры, состоит из 4 колонок:

- поле id_face - типа serial, авто заполняемый уникальный идентификатором лицевого параметра, является первичным ключом таблицы;
- поле file – типа bytea, хранит файл фотографии из которого сформировались лицевые параметры;
- поле face_data – типа bytea, хранит лицевые параметры;
- поле personal_id – типа serial, хранит ссылку на работника персонала, которому принадлежат данные лицевые параметры.

Таблица «cabinets» предназначена для хранения списка кабинетов, состоит из 4 колонок:

- поле id_cab - типа serial, авто заполняемый уникальный идентификатором кабинета, является первичным ключом таблицы;
- поле name – типа text, предназначена для хранения названия кабинета;
- поле floor – типа int, хранит номер этажа, на котором расположен кабинет;
- поле dep_id – типа serial, хранит ссылку на отдел, которому принадлежит кабинет.

Таблица «cameras» предназначена для хранения списка камер, состоит из 5 колонок:

- поле id - типа serial, авто заполняемый уникальный идентификатором камеры, является первичным ключом таблицы;
- поле cam_model – типа text, предназначен для хранения названия камеры;
- поле addr – типа ipnet, хранит ip address, по которому доступна камера;
- поле cab_id – типа serial, хранит ссылку на кабинет, к которому приписана камера;

- поле `in_pos` – типа `bool`, в значении `True`, если камера расположена на вход в кабинет;
- поле `status` - типа `bool`, хранит значение параметра работоспособности камеры.

Таблица «`in_out_date`» предназначена для фиксирования информации, о входящих и выходящих сотрудников, состоит из 4 колонок:

- поле `time` – типа `TIMESTAMPZ`, авто заполняемая дата и время, когда опознали лицо;
- поле `per_id` – типа `serial`, хранит ссылку на работника персонала, которому принадлежат данные определённого лица;
- поле `cab_id` – типа `serial`, хранит ссылку на кабинет, к которому приписана камера, определившая лицо;
- поле `cam_id` – типа `serial`, хранит ссылку на камеру, к которой приписана камера;
- поле `direction` - типа `bool`, хранит положительное значение, если сотрудник заходит в кабинет.

Так же в PostgreSQL созданы несколько ролей, обладающими различными права:

- поле `admin` - обладает всеми возможностями для работы с базой данных;
- поле `reader_all` - обладает только возможностями чтения данных из всех таблиц;
- поле `reader_basic` - обладает только возможностями чтения всех таблиц, кроме таблиц `faces`, `department`, `personal`.

3.3 Разработка программного обеспечения системы

Вначале программного кода сервера обработчика, состоящего из одного файла, идёт включение необходимых библиотек для функционирования приложения, а именно:

- библиотека «OpenCV»;
- библиотека «face_recognition»;
- библиотека «Thread»;
- библиотека «datetime»;
- библиотека «sys».

Далее представлено описание главного класса, который будет отвечать за обработку видеопотоков с камеры - `ThreadedCamera`. Класс имеет следующие внутренние переменные:

- переменная `known_faces` - внутренний массив биометрических данных лиц, которые загружаются из базы данных;
- переменная `capture` - объект, представляющий интерфейс для взаимодействия с захватываемым видеопотоком;
- переменная `frame` - хранит, считываемый из потока;
- переменная `threadingFind` - объект, отвечающий за запуск в отдельном потоке метода `find_face`;
- переменная `thread` - объект, отвечающий за запуск в отдельном потоке метода `update`;
- переменная `thread_send` - объект, отвечающий за запуск в отдельном потоке метода `send_faces`;
- переменная `founded_faces` - массив, хранищий список биометрических данных лиц, обнаруженных на кадре;
- переменная `db_connector` - объект, предоставляющий интерфейс, для работы с базой данных (загрузка новых лиц, отправка данных в базу данных)ю

Класс состоит из следующих методов:

- метод `init` – конструктор класса, создаётся при объявлении переменной класса, в этом методе реализовано объявление переменных необходимых для создания видеопоток (хранящие данные видеопотока), хранения данных о параметрах лиц, так же там идёт инициализация дополнительных программных потоков, для увеличения производительности системы, а так же запуск одного из потоков (поток считывания изображения с видеопотока камеры);
- метод `update` - метод, в котором происходит считывания видео потока, считывания кадров после запуска видео потока, а также запуск программного потока для анализа лиц с кадров, данный метод и запускается в отдельном программном потоке;
- метод `show_frame` - предназначен для показа обработанных кадров в окне;
- метод `find_face` - метод запускаемый в отдельном поток, назначение данного поток состоит в считывании копии текущего кадра, а затем его обработка, для дальнейшего его анализа на поиск лиц в кадре (сжатие изображения), при обнаружении лиц в кадре, система анализирует лица со списком сохранённых лиц в памяти, при наличии данных лиц во внутреннем массиве `known_faces` запуска;
- метод `update_faces` - метод, также работающий в отдельном потоке и состоящий из бесконечного цикла, который реализует обновление внутреннего массива `known_faces` раз в сутки, данный метод реализован без

использования триггеров, т.к. триггеры не запускают функции в отдельных потоках;

- метод `send_faces` - метод, использующий `db_connector`, чтобы отправить в базу данных информацию об опознанных лицах или о лицах, чьи лица не находятся во внутреннем массиве.

После описание класса идёт описание главной функции программы `main`. В неё в начале идёт инициализация объекта класса `ThreadedCamera` с предоставлением ему в конструктор аргумента, содержащего адрес видео поток камеры, далее идёт бесконечный цикл, в котором в обработчике исключений вызывается метод `show_frame`, класса `ThreadedCamera`. При прерывании цикла вызывается функция `destroyAllWindows` библиотеки «OpenCV», которая закрывает все окна.

Весь программный код сервера-обработчика можно рассмотреть в приложении А.

Программный код сервера веб-интерфейса построен на использовании фреймворка «Django», который представляет собой набор реализующих автоматическое создание таблиц в базе данных PostgreSQL, так же позволяет отправлять данные в таблицы, напрямую не используя коннектор базы данных. Однако выше описанный программный код лишь формирует «backend» часть, то есть лишь предоставляет `api`, которое необходимо задействовать, чтобы на веб-странице проекта отображалась информация.

Часть веб-интерфейса «frontend» пользовательского интерфейса представлена в виде 3 файлов:

- файл `index.html` -отвечающий за расположение элементов интерфейса страницы;
- файл `style.css` – отвечающий за внешнее оформление различных частей интерфейса страницы;
- файл `app.js` – отвечающий за связывание разделов страницы, отвечающих за отображение информации для пользователя, с данными, которые возвращают `api`-запросы, отправленные на «backend».

Административная часть не требует отдельной разработки схожих файлов.

На рисунке 14 и листе графического материала ГУИР.466152.002 ПД изображена диаграмма классов, отражающая взаимодействие программных классов веб-интерфейса между собой.

Программный код веб-интерфейса приведён в приложении А.

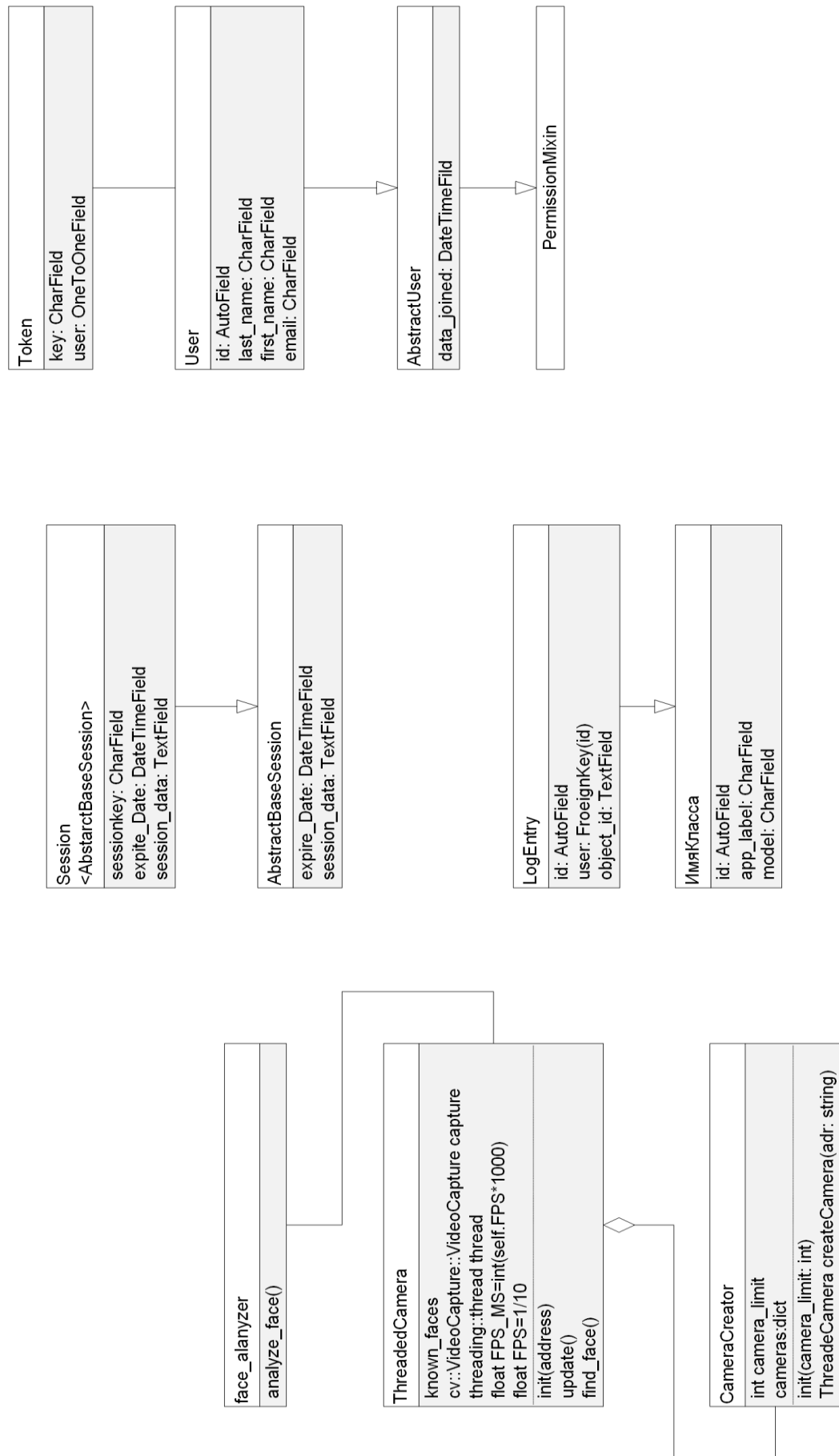


Рисунок 14 – Диаграмма классов

3.4 Проектирование последовательности развёртывания системы

Развёртывание системы подразделяется на 3 этапа: подготовка, монтаж и установка с настройкой.

Первый этап включает себя действия, начинающиеся от планирования использования и до этапа получения закупленного оборудования.

В первую очередь создаётся план, в котором должно указываться следующая информация:

- список кабинетов, требующих использования данной системы;
- список персонала, а также их фотографии, для дальнейшей их обработки;
- схема сетевой топологии системы, содержащий все элементы данной системы, в схеме должны быть отображены связи этих элементов, а также их минимальные настройки;
- составление списка оборудования для системы;
- составление списка денежных затрат на монтаж системы.

В список оборудования для закупки обязательно должны включаться:

- камеры;
- кабель LAN;
- внешние кабельные каналы;
- маршрутизаторы;
- сервер обработчик.

Далее происходит согласование плана, после чего происходит закупка оборудования, а также поиск подрядных организаций для монтажа системы, если отсутствуют возможности монтажа собственными силами организации.

Этап монтажа включает в себя:

- создание кабельных каналов для прокладки цепей связи между оборудованием системы;
- монтаж устройств ограничения физического доступа для камер и помещений, содержащих в себе элементы системы;
- прокладка дополнительных питающих электрических цепей, при отсутствии или нехватки их;
- прокладка кабельных линий связи;
- монтаж камер.

Этап установки и настройки подразумевает настройка программного обеспечения и сетевого окружения системы. А именно:

- настройка видео потока с камер;
- настройка сетевых настроек операционной системы;

- запуск контейнеров, в которых буду производиться работа частей системы.

Вначале необходимо соединить камеры, а также сервер разработчик в одну локальную сеть, либо напрямую, либо используя маршрутизаторы.

Первоначально на сервере обработчике необходимо установить операционную систему Ubuntu 20.04, при необходимости или нежелании пользоваться графическим интерфейсом операционной системы можно установить серверную вариацию.

На установленной операционной системе необходимо установить следующий набор программ:

- Docker;
- Docker-compose;
- OpenSSH-server.

Далее нужно на сервер обработчик перенести весь программный проект. Затем в терминале операционной системе нужно запустить Docker compose, указав путь до docker-compose.yml. После этого нужно следить за docker контейнерами, которые запускаются в система. После того как все контейнеры запустятся, можно отключить серверу-обработчику доступ к интернету, т.к. все нужные пакеты данных были скачаны.

На рисунке 15 и листе графического материала ГУИР.466152.004 ПД рассмотрена диаграмма развёртывания, дающая схематичное представление о ходе развёртывания программного обеспечения на сервере-обработчике и сервере базы данных и веб-интерфейса.

Порядок развёртывания сервера обработчика:

- запуск программы Docker с помощью файла Dockerfile;
- создание контейнера и копирование в него файлов, содержащих программный код сервера обработчика.

Порядок развёртывания веб-сервера:

- запуска файла runs.sh;
- запуск docker-composer с использованием файла docker-compose.yml;
- создание контейнера с базой данных PostgreSQL;
- создание контейнера с программный кодом фреймворка «Django» и копирование в него программного кода веб-интерфейса части «backend»;
- создание контейнера с Nginx;
- создание контейнера с Redis.

После развёртывания системы необходимо наполнение базы данных, с использованием административной части веб-интерфейса.

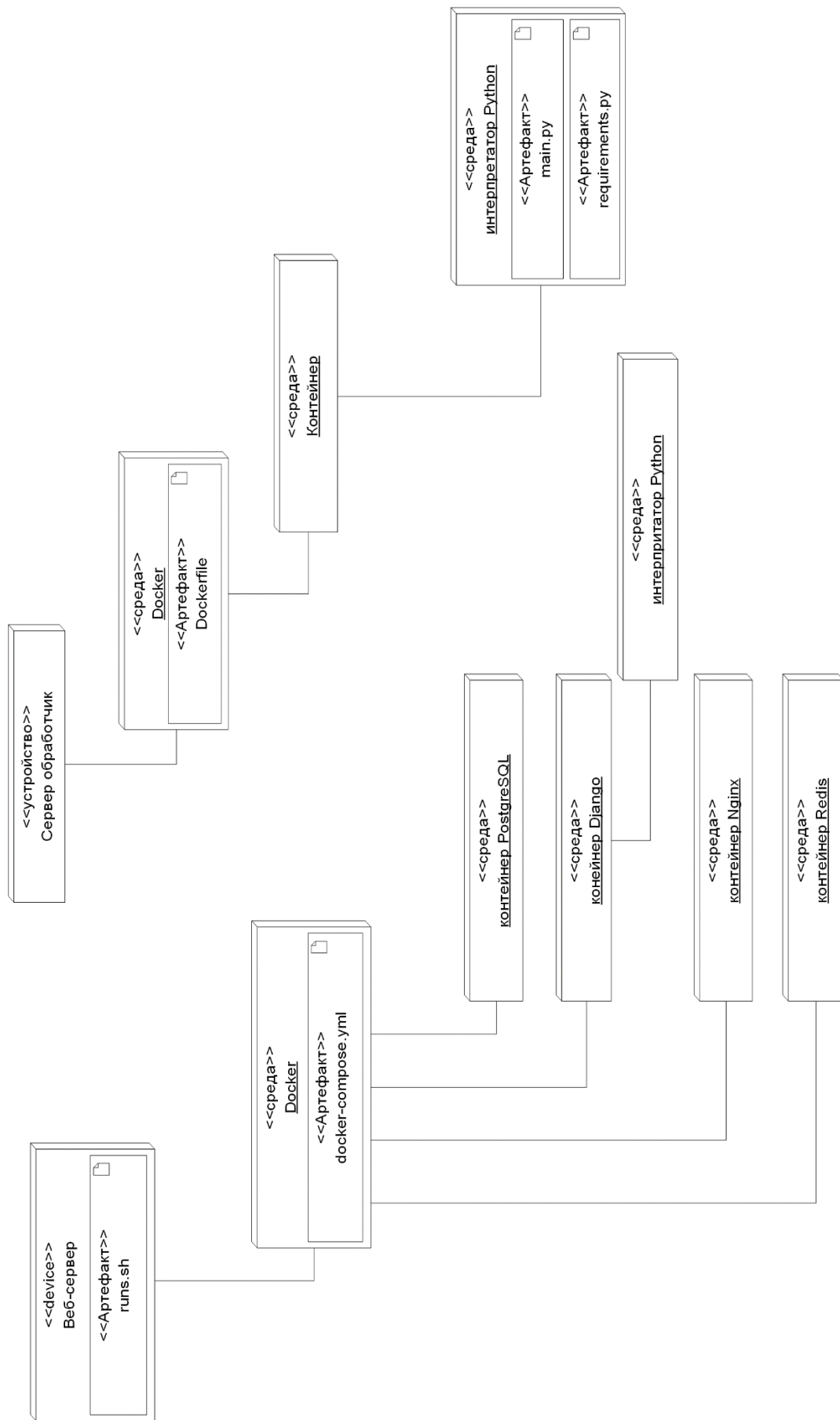


Рисунок 15 – Диаграмма развёртывания

3.5 Экранные формы веб-интерфейса

Веб-интерфейс по уровню допуска подразделяется на 2 части:

- пользовательская часть;
- административная часть.

Пользовательская часть предназначена для предоставления свободного доступа к части таблиц в базе данных проекта, а именно:

- список персонала - раздел «Personal Info»;
- список кабинетов - раздел «Cabinets status»;
- список камер - раздел «Camera status»;
- список посещения персонала кабинетов - раздел «In-Out status».

Более подробная информация, о возможности, пользовательской части предоставлена в схеме вариантов использования графической в списке графических документов. Далее подробнее рассмотрены элементы графического листа ГУИР.466152.007 ПД.

Страница содержит следующие элементы:

- «шапка» - верхняя часть страницы (см. рисунок 16), содержащая название открытого раздела;



Рисунок 16 – «Шапка» страницы

- навигационная панель (см. рисунок 17), предназначенная для переключения между разделами;

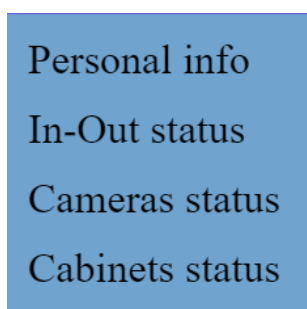


Рисунок 17 – Навигационная панель

- кнопка «Login» - находящаяся в верхнем правом углу страницы (см. рисунок 18) и открывающая окно авторизации для входа в административную часть.

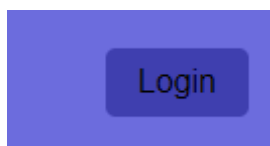


Рисунок 18 – Кнопка авторизации административного доступа

Остальная часть страницы предназначена для отображения разделов страницы. Далее описывается отображение отделов страницы.

Раздел «Personal info» - предназначен для отображения списка персонала (см. рисунок 19), список представлен в виде таблицы с 3 колонками:

- колонка «id pers» - идентификационный номер работника;
- колонка «name» - ФИО сотрудника;
- колонка «dep id» - название отдела, в котором работает сотрудник.

Так же этот раздел содержит окно поиска персонала по имени и/или идентификационному номеру в базе данных (см. рисунок 20).

id pers	name	dep id
1	kir lname fname	dep1
2	user1 lname fname	dep3
3	kir lname fname	dep3

Рисунок 19 – Список персонала

Рисунок 20 – Окно поиска

Раздел «In-Out status» - предназначен для отображения регистрации посещений помещений в виде таблицы (см. рисунок 21) со следующими столбцами:

- столбец «id» - идентификационный номер посещения;
- столбец «timedate» - дата и время посещения;
- столбец «direction» - направление, красное – из кабинета, зелёное – в кабинет;
- столбец «per id» - идентификационный номер опознанной персоны;
- столбец «cab id» - идентификационный номер кабинета, где зафиксировалось событие;
- столбец «cam id» - идентификационный номер камеры, зафиксировавшей лицо.

id	timedate	direction	per id	cab id	cam id
1	2022-12-28T11:50:08.688797Z	●	1	1	1
2	2022-12-28T11:50:19.978811Z	●	1	1	2

Рисунок 21 – Список посещения помещений

Раздел «Cameras status» предназначен для отображения списка камер, задействованных в системе, в виде таблицы (см. рисунок 22) со следующими столбцами:

- столбец «id cam» - идентификационный номер камеры;
- столбец «cam model» - название модели камеры;
- столбец «cab id» - идентификационный номер кабинета, в котором задействована камера;
- столбец «in pos» - позиция камеры, относительно направления входа в кабинет: зеленый цвет – в кабинет, красный цвет – из кабинета.

Так же этот раздел содержит окно поиска камер по модели и/или идентификационному номеру в базе данных, окно полностью повторяет окно поиска персонала (см. рисунок 20).

id cam	cam model	cab id	in pos
1	model1	cab1	●
2	cam2	cab1	●

Рисунок 22 – Список камер

Раздел «Cabinets status» предназначен для отображения списка помещений, представленного в виде таблицы (см. рисунок 23) со следующими столбцами:

- столбец «id cab» - идентификатор помещения;
- столбец «name» - название кабинета;
- столбец «floor» - номер этажа, где расположено помещение;
- столбец «dep id» - идентификатор отдела, которому принадлежит сообщение.

Так же этот раздел содержит окно поиска помещений по названию и/или идентификационному номеру в базе данных, окно полностью повторяет окно поиска персонала (см. рисунок 20).

id cab	name	floor	dep id
1	cab1	1	dep1
2	cab2	2	dep2
3	cab3	3	dep3
4	cab4	9	dep2

Рисунок 23 – Список камер

Административная часть веб-интерфейса вначале представляет собой окно авторизации (см. рисунок 24).

После прохождения авторизации, открывается главная страница администратора (см. рисунок 25), оно предназначено для организации доступа к базе данных, а также его заполнения.

Она представлена в виде нескольких разделов:

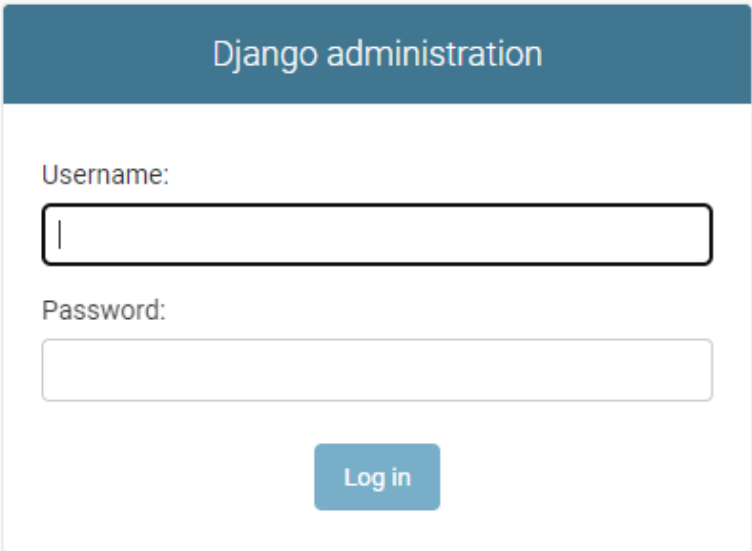
- раздел «AUTH TOKEN» - раздел создания ключей доступа, для связи базы данных с другими системами;
- раздел «AUTHENTICATION AND AUTHORIZATION» - раздел для создания пользователей базы данных и объединения их в группы;
- раздел «CORE» - предназначен для работы с таблицами базы данных, а именно: просмотр данных, редактирование, добавление новых данных в таблицы.

Каждый из перечисленных разделов содержит наборы таблиц, строки в которых можно редактировать либо добавлять новые.

На рисунке 26 представлен пример страницы работы с таблицей «cabinets». На которой представлен список существующих строк в таблице,

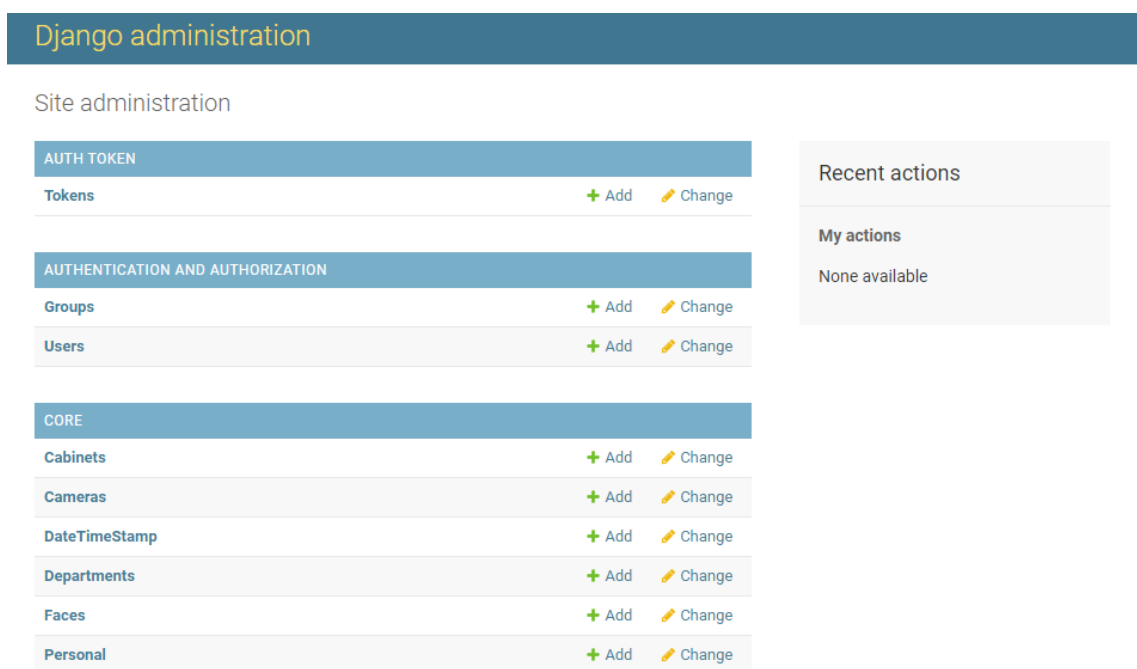
окно поиска, которое производит поиск по всем столбцам таблицы, кнопка добавления новой строки в таблицу.

На рисунке 27 представлена страница добавления новой строки в таблицу, на примере добавления новой строки в таблицу «Cabinets».



The image shows the Django administration login interface. It features a dark blue header with the text "Django administration". Below the header, there are two input fields: "Username:" and "Password:". The "Username:" field is a white box with a black border and a cursor. The "Password:" field is a white box with a black border. Below the password field is a blue button with the text "Log in".

Рисунок 24 – Окно авторизации администратора



The image shows the Django administration main page. It has a dark blue header with the text "Django administration". Below the header, there is a section titled "Site administration". Under this section, there are three main categories: "AUTH TOKEN", "AUTHENTICATION AND AUTHORIZATION", and "CORE". Each category has a list of items with "Add" and "Change" links. The "AUTH TOKEN" category has one item, "Tokens". The "AUTHENTICATION AND AUTHORIZATION" category has two items, "Groups" and "Users". The "CORE" category has seven items: "Cabinets", "Cameras", "DateTimeStamp", "Departments", "Faces", and "Personal". To the right of the "Site administration" section, there is a "Recent actions" section with a "My actions" subsection. The "My actions" subsection shows "None available".

AUTH TOKEN	
Tokens	+ Add Change

AUTHENTICATION AND AUTHORIZATION	
Groups	+ Add Change
Users	+ Add Change

CORE	
Cabinets	+ Add Change
Cameras	+ Add Change
DateTimeStamp	+ Add Change
Departments	+ Add Change
Faces	+ Add Change
Personal	+ Add Change

Recent actions

My actions

None available

Рисунок 25 – Главная страница администратора

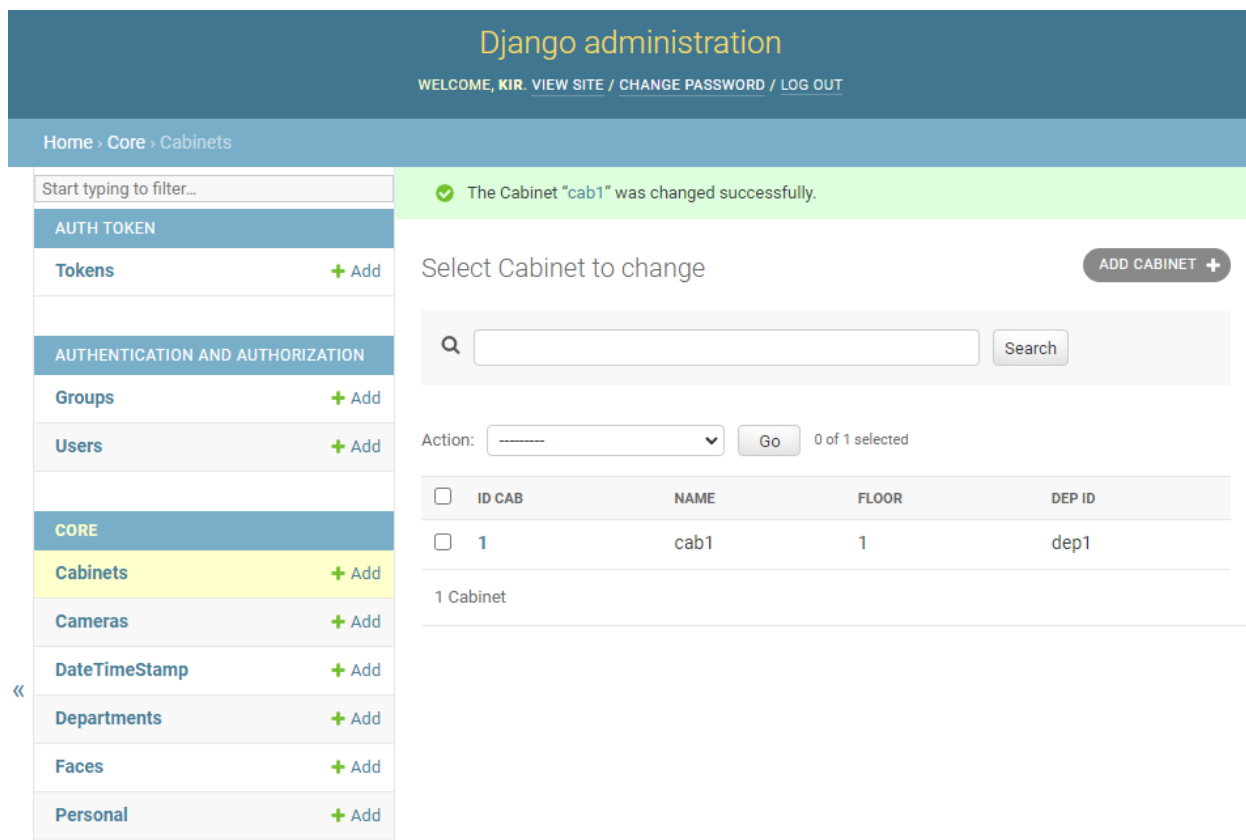


Рисунок 26 – Страница таблицы «Cabinets»

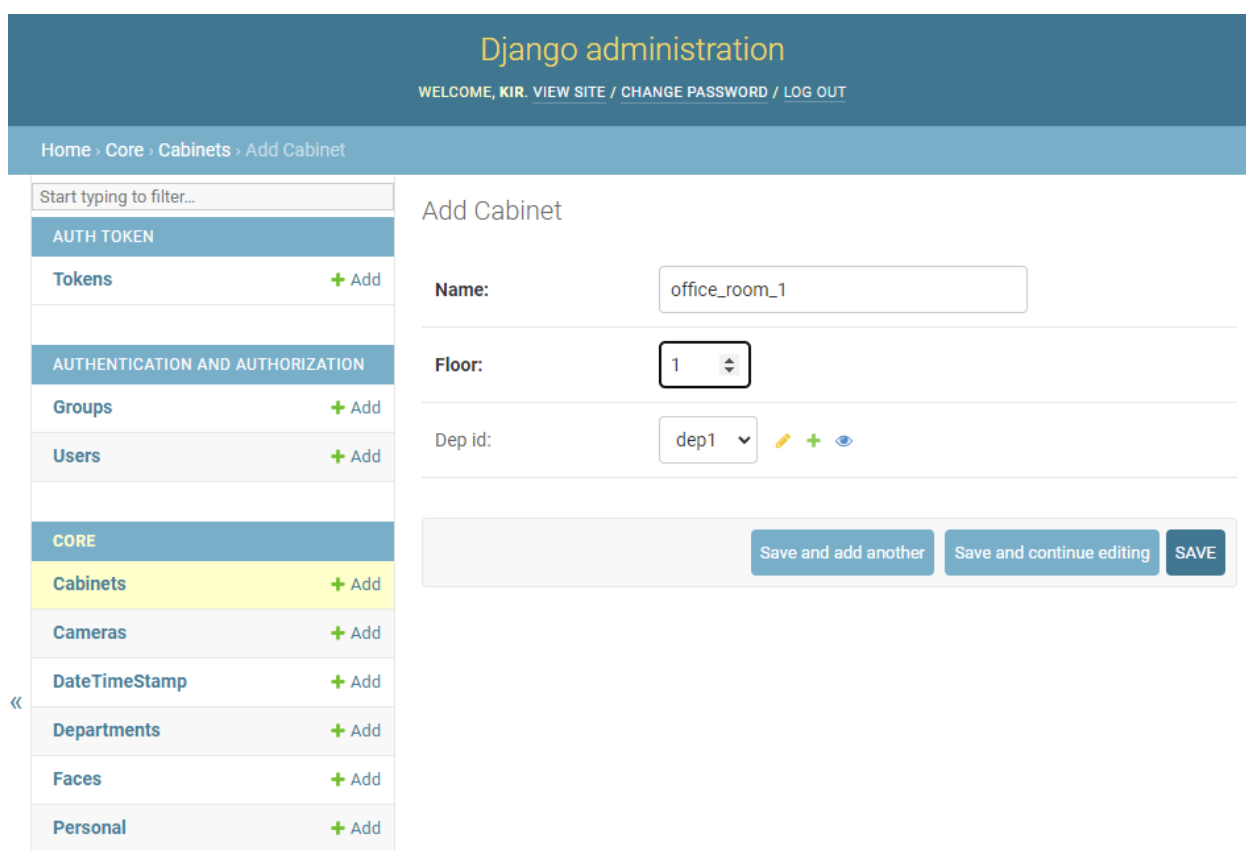


Рисунок 27 – Пример страницы добавления строки в таблицу «Cabinets»

На рисунке 28 пример того, как в базе данных сохраняется кадр с неопознанным лицом.

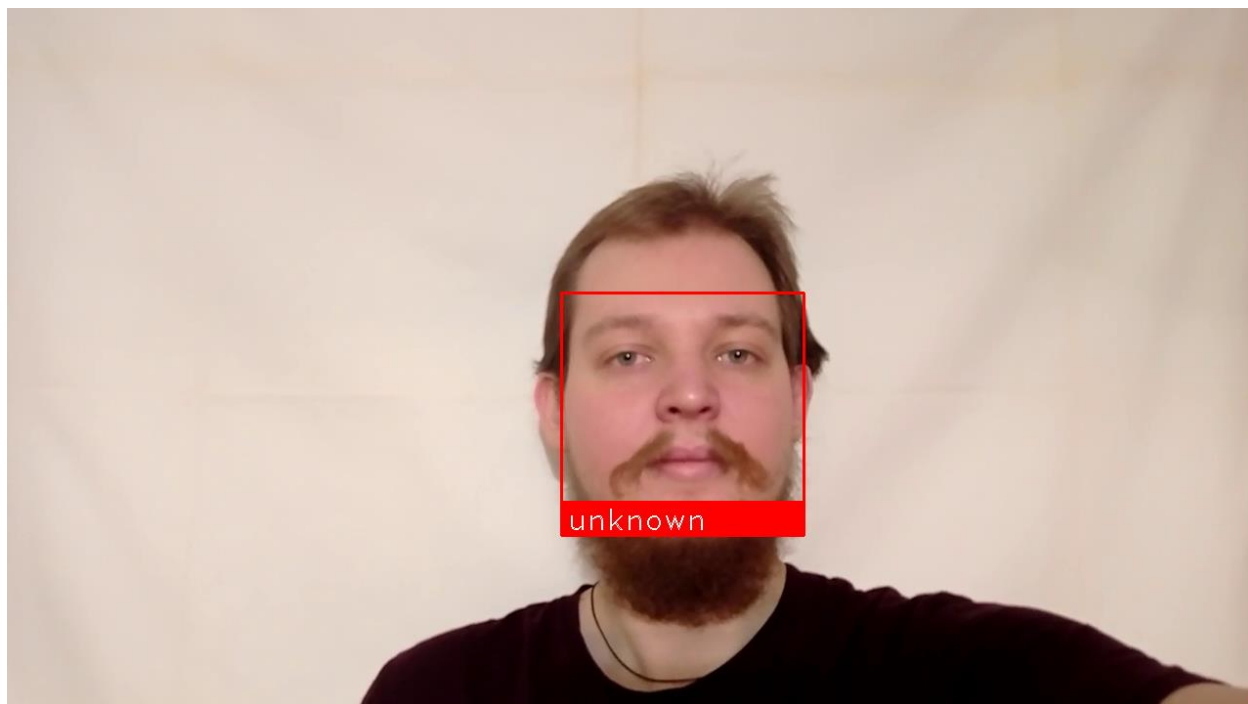


Рисунок 28 – Пример изображения с неопознанным лицом

Вывод: Система состоит из двух частей - веб-интерфейс и обработчик-камеры; веб-интерфейс реализован с помощью фреймворка «Django», реализующего frontend и backend части веб-интерфейса. Разработана диаграмма базы данных, описывающая структуру базы данных разрабатываемой системы. Обработчик камер реализован с помощью языка программирования Python и библиотек «OpenCV», «face_recognition». Алгоритм работы обработки камер состоит нескольких потоков: получение кадра из видеопотока камеры, нахождение лица на кадре, сравнение лица с лицами из базы данных. Разработана диаграмма развёртывания, иллюстрирующая порядок развёртывания программной части системы. Описаны экранные формы веб-интерфейса системы.

4. НАСТРОЙКА КЛЮЧЕВЫХ СВОЙСТВ АВТОМАТИЧЕСКОЙ СИСТЕМЫ УЧЁТА ПЕРСОНАЛА

4.1 Конфигурирование параметров безопасности системы

Безопасность системы разделяется на 3 физических, сетевой, программный.

Реализация безопасности системы на физическом уровне заключается в осуществлении следующих пунктов:

- ограничен физический доступа в помещение, где расположен сервер-обработчик, путём запирающих устройств, ключи к которым находятся только у системного администратора, начальника отдела, курирующего данную систему;

- ограничен физический доступ к промежуточному сетевому оборудованию системы (коммутаторы, маршрутизаторы и т.п.), путём размещения их в помещениях/распределительных коробах, имеющих системы запирания и ключевой доступ;

- ограничен физический доступа к узлам подключения и настройки камер, путём монтажа, не допускающего доступ, без использования специализированных инструментов – внутренняя проводка;

- ограничен физический доступа к узлам питания всех элементов системы – прокладка линий питания напрямую в щиты электропитания, которые в свою очередь имеют запирающие устройства.

Сетевой уровень безопасности подразумевает отключения возможностей доступа к серверу-обработчику не через http порт, а также доступа к ip адресам камер и промежуточного оборудования, для этого выполняются следующие действия:

- у коммутаторов отключено использование не задействованных физических портов;

- включен фаервол на серверах-обработчиках, а также на сервере веб-интерфейса и базы данных;

- добавления в реестр разрешённых ip адресов – на сервере обработчике в реестр добавляются только ip адреса камер, с которых получает данные сервер, а также свейанного с ним сервера веб-интерфейса, также добавлен один ip адрес компьютера системного администратора;

- при развёртывании контейнеров в docker в файлах Dockerfile контейнеров в поле «ports» указывается ip адрес - 127.0.0.1, чтобы к

контейнерам был доступ только с хостинговой машины, кроме контейнера отвечающего за web часть системы;

- у промежуточного сетевого оборудования включено использование «white-list» ip адресов, в этот же список добавить только ip адреса устройств, используемых в системе;

- на сервере-обработчике для контейнеров, обрабатывающих данные с камер создана отдельная внутренняя сеть в Docker, на сервере веб-интерфейса для контейнеров тоже создана отдельная подсеть в Docker.

Программный уровень безопасности заключается в создании 2 учётных записей для веб-интерфейса приложения. Это реализовано путём использования создание 2ух уровней доступа через фреймворк Django. Базовый уровень - позволяет:

- посмотреть список персонала;
- найти конкретного представителя персонала отдела;
- посмотреть данные о входящем и исходящем персонале;
- посмотреть список камер и информацию о них;
- посмотреть список кабинетов.

Административный уровень предоставляет полный доступ к базе данных системы.

Так же в контейнерах, которые обрабатывают изображения с камер, будет использоваться специально созданный пользователь в базе данных, чтобы контейнер имел доступ только к необходимой информации. Данный пользователь будет обладать только следующими разрешениями: чтение списка биометрии лиц, добавление информации, об опознанном лице.

Так же благодаря использованию фреймворка, исключена возможность атак с использованием SQL-инъекций.

4.2 Проектирование вариантов масштабируемости и интеграции системы

Масштабируемость, в электронике и информатике, означает способность системы, сети или процесса справляться с увеличением рабочей нагрузки (увеличивать свою производительность) при добавлении ресурсов (обычно аппаратных). Масштабируемость – важный аспект электронных систем, программных комплексов, систем баз данных, маршрутизаторов, сетей и т. п., если для них требуется возможность работать под большой нагрузкой. Система называется масштабируемой, если она способна увеличивать производительность пропорционально дополнительным

ресурсам. Масштабируемость можно оценить через отношение прироста производительности системы к приросту используемых ресурсов. Чем ближе это отношение к единице, тем лучше. Также под масштабируемостью понимается возможность наращивания дополнительных ресурсов без структурных изменений центрального узла системы.

В системе с плохой масштабируемостью добавление ресурсов приводит лишь к незначительному повышению производительности, а с некоторого «порогового» момента добавление ресурсов не даёт никакого полезного эффекта.

Она может быть:

- вертикальной - увеличение производительности каждого компонента системы с целью повышения общей производительности. Масштабируемость в этом контексте означает возможность заменять в существующей вычислительной системе компоненты более мощными и быстрыми по мере роста требований и развития технологий. Это самый простой способ масштабирования, так как не требует никаких изменений в прикладных программах, работающих на таких системах;

- горизонтальной - разбиение системы на более мелкие структурные компоненты и разнесение их по отдельным физическим машинам (или их группам), и (или) увеличение количества серверов, параллельно выполняющих одну и ту же функцию. Масштабируемость в этом контексте означает возможность добавлять к системе новые узлы, серверы для увеличения общей производительности. Этот способ масштабирования может требовать внесения изменений в программы, чтобы программы могли в полной мере пользоваться возросшим количеством ресурсов.

В качестве вертикального масштабирования используется замена вычислительных мощностей, на более производительные аналоги. А именно, замена материнской платы сервера, замена центрального процессора, замена оперативной памяти. Замена одного их перечисленных компонентов сервера может повлечь замену других компонентов.

Например, замена оперативной памяти может потребовать замену материнской платы, т.к. новая оперативная память может обладать более высокой частотой памяти, нежели та частота, которую может поддерживать материнская плата. На рисунке 28 представлена разница частот оперативной памяти 3-его поколения, что показывает различные стандарты частот памяти в одном поколении.

То есть при замене памяти типа DDR3-1600 на тип DDR3-2400, текущая материнская плата может не поддерживать новый тип.

Стандартное название	Частота памяти, МГц	Время цикла, нс	Частота шины, МГц	Эффективная (удвоенная) скорость, млн. передач/с	Название модуля	Пиковая скорость передачи данных при 64-битной шине данных в одноканальном режиме, МБайт/с
DDR3-800	100	10,00	400	800	PC3-6400	6400
DDR3-1066	133	7,50	533	1066	PC3-8500	8533
DDR3-1333	166	6,00	667	1333	PC3-10600	10667
DDR3-1600	200	5,00	800	1600	PC3-12800	12800
DDR3-1866	233	4,29	933	1866	PC3-14900	14933
DDR3-2133	266	3,75	1066	2133	PC3-17000	17066
DDR3-2400	300	3,33	1200	2400	PC3-19200	19200

Рисунок 29 – Различия частот оперативной памяти третьего поколения (DDR3)

Так же схожая ситуация возникает, при решении, о замене оперативной памяти на более новое поколение, т.к. с каждым новым поколением частота памяти сильно возрастает [20]. В таблице 2 приведены технические параметры 4 поколений оперативной памяти, как с увеличением номера поколения – возрастают и границы минимальных и максимальных частот памяти в рамках одного поколения.

Таблица 2 – Различия параметров оперативной памяти различных поколений оперативной памяти.

Поколение	Частота, МГц	Количество контактов	Напряжение питания, В	Скорость передачи (Гб/с)
DDR	266-400	168	2.5	2.1-3.2
DDR2	533-800	184	1.8	4.2-6.4
DDR3	1066-1600	240	1.35	8.5-14.9
DDR4	2133-3200	288	1.2	17-21.3

Аналогичная ситуация складывается и при решении, о замене центрального процессора на более производительный. В разрабатываемом проекте ключевыми параметрами для процессора являются:

- максимальная частота – от данного параметра зависит скорость обработки данных, получаемых от камеры;

- максимальное количество потоков – параметр, который в проекте накладывает ограничения на количество обрабатываемых камер.

Т.к. максимальное число потоков процессора напрямую зависит от количества ядер процессора, то соответственно при росте числа ядер – возрастает число поддерживаемых потоков, что напрямую сказывается на производительности процессора. На рисунке 29 мы можем видеть график роста производительности процессора при возрастании числа его ядер

Оба параметра взаимосвязаны, т.к. даже если заменять процессор на более производительный, но имеющий то же число максимальных потоков, то общая производительность системы не особо возрастёт. Однако, при замене, новый процессор выбирать исходя из увеличения количества поддерживаемых потоков [21].

В разрабатываемой системе горизонтальным масштабированием является увеличение количество обрабатываемых камер. А т.к. обработка каждой камеры происходит в отдельно развёрнутом контейнере, то кроме физического увеличения числа камер в системе, для каждой добавленной камеры будет добавлен контейнер-обработчик.

Для более наглядного понимания процесса горизонтального масштабирования нужно понимать диаграмму компонентов, представленную в списке графического материала.

Но при горизонтальном масштабировании не стоит забывать о вертикальном. Т.к. при добавлении контейнеров возрастает нагрузка на сервер, то стоит учитывать, что возможно придётся заняться увеличением производительности сервера, чем и является вертикальное масштабирование.

Интеграция данной системы затрагивает множество спектров работы предприятия, а т.к. система помогает решать задачи в двух сферах (безопасности и учёте рабочего времени сотрудников), то она затрагивает все спектры функционирования предприятия, но также это зависит от целей, ради которых внедряется данная система. Далее в данном пункте об интеграции системы будет рассмотрена исходя из этих целей.

Изначально будет рассматриваться сферы, в которых будет происходить интеграция, которые будут затрагиваться независимо от целей и имеющие общие черты.

Независимо от целей введения системы, потребуется интеграция системы в компьютерную сеть предприятия, это необходимо для осуществления доступа к базе данных системы с не хостинговой машины. Так

же, системе нужен будет доступ в интернет, для скачивания необходимого программного обеспечения, но этот доступ необходим только на этапе установки и наладки система, а также при необходимости обновления программного обеспечения системы.

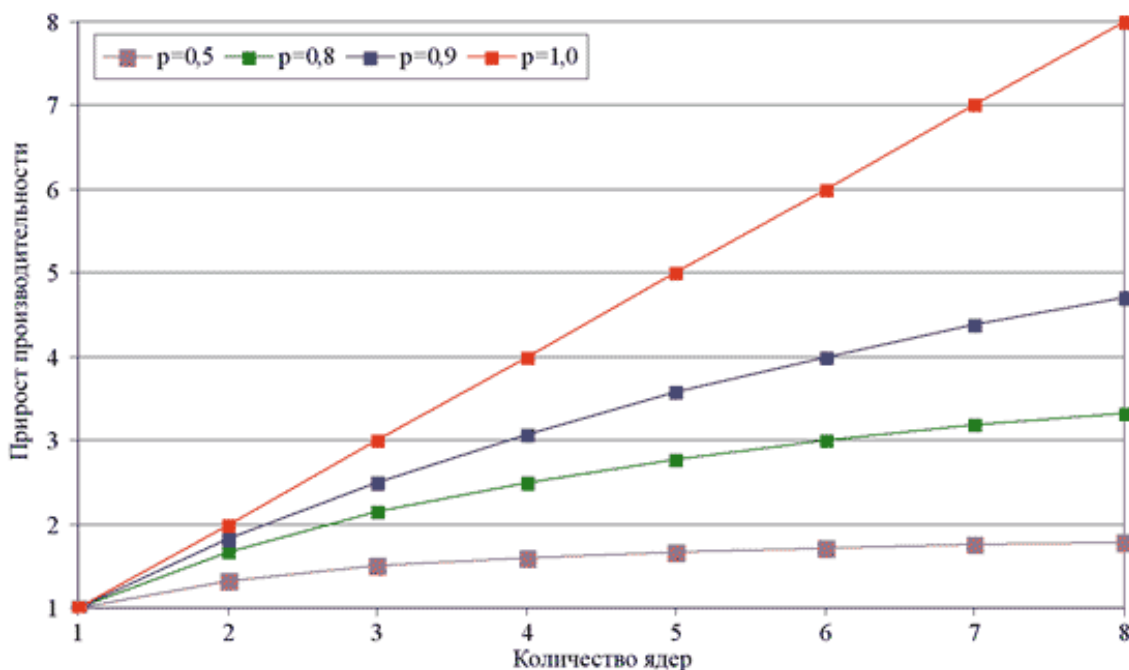


Рисунок 29 – График зависимости прироста производительности от числа ядер процессора.

Необходимым является создание пользовательских учетных записей в базе данных для осуществления системы контролируемого доступа. Самым важным является добавление в базу данных системы изображений допущенных сотрудников.

При введении системы ради целей безопасности, то важным является интеграция системы в систему безопасности предприятия, это осуществляется разработкой программы, сканирующей базу данных системы на наличии людей в помещении, а также создание ролей для доступа в базу данных, обладающей необходимыми полномочиями (например, добавление записи, что помещение очищено, когда не сработала фиксация лица, вышедшего из помещения).

Так же, если необходимо использовать систему для использования её базы данных, для осуществления доступа в помещения, но интеграция заключается лишь в добавлении специализированной роли в базу данных системы, а также осуществить связь по сети предприятия между системами.

Если же система необходима для учета рабочего времени сотрудников, то для этого необходима интеграция базы данных системы в систему учета рабочего времени сотрудников, для чего необходимо создать в базе данных специализированную роль\и, а в системе учета рабочего времени лишь использовать «коннектор» подходящий под базу данных вводимой системы. Так же для этого можно использовать арі-запросы, позволяющие получить ту же информацию, что присутствует в веб-интерфейсе.

Вывод: Безопасность системы строится на использование встроенного функционала используемого оборудования и программного обеспечения. Для безопасности системы ограничивается доступ по каналам связи, не используемых системой, также создание отдельных подсетей внутри общей сети системы позволяет реализовать разграничение доступа к различным сетевым компонентам системы. Система поддерживает вертикальное и горизонтальное масштабирование. Горизонтальное выражено в виде увеличения количества контейнеров, обрабатывающих сигналы с камер, и увеличении числа серверов-обработчиков. Вертикальное масштабирование заключается в повышении вычислительных мощностей оборудования, а именно серверов-обработчиков и сервера базы данных и веб-интерфейса. Вертикальное масштабирование выражено в виде замены комплектующих серверов на более производительные варианты. Разрабатываемая система может быть интегрирована в другие систем предприятия, путём создания отдельных профиле доступа в базе данных системы и обеспечения физического доступа к базе данных.