3. ВЫБОР ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И ОБОРУДОВАНИЯ ДЛЯ РАЗРАБОТКИ СИСТЕМЫ

3.1 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 сочетания 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_recogniton» - 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 низка 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].

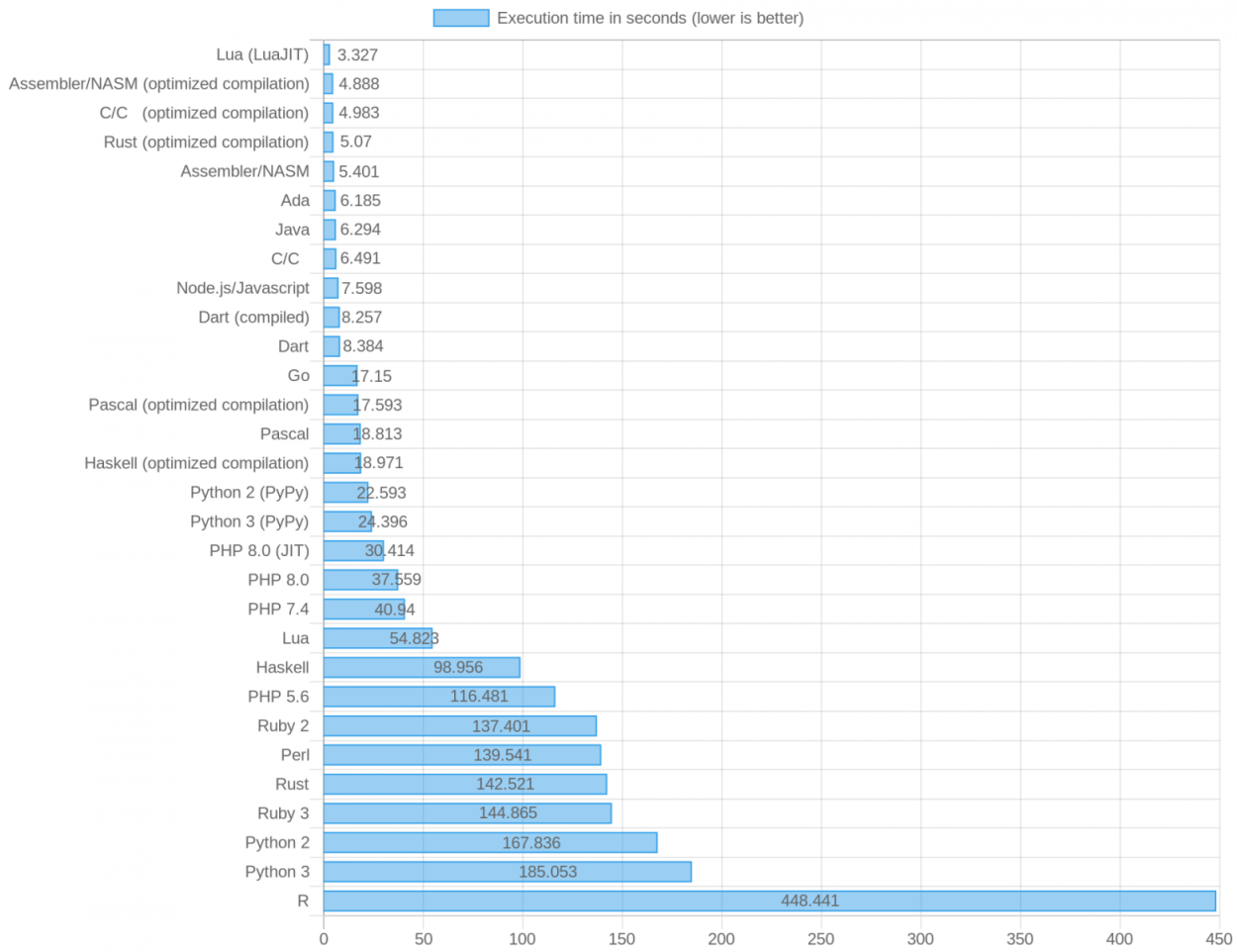

Рисунок 6 – График скорости выполнения теста на основе перебора простых чисел для различных языков программирования

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.

3.2 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].

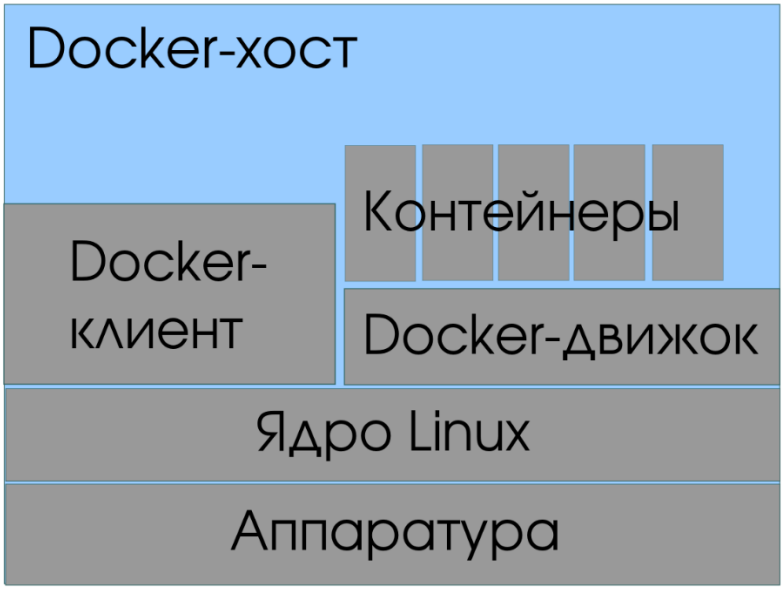

Рисунок 7 – Docker на физическом Linux-сервере

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», «Mercurical»;
- 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 мультиплексирования 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.

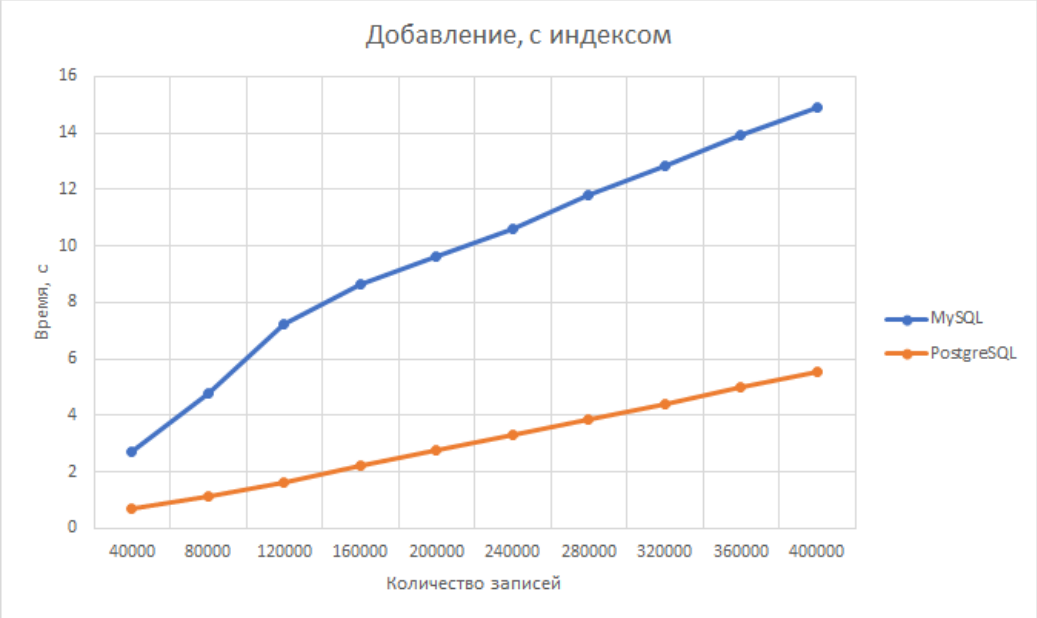

Рисунок 8 – График производительности, при выполнении добавлении записей в таблицу.

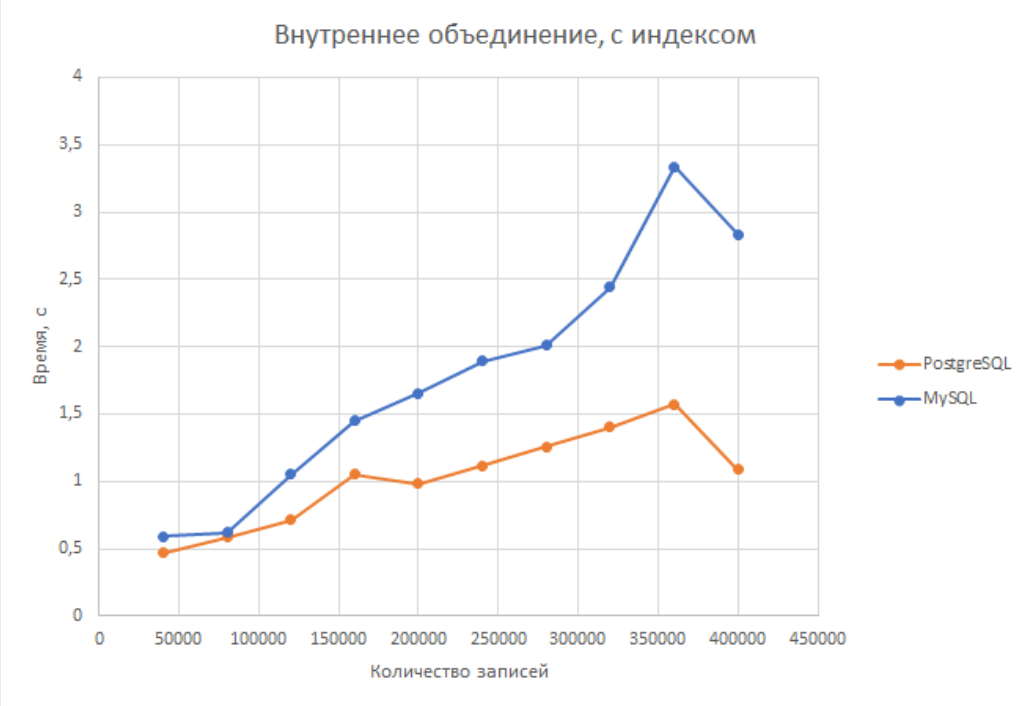

Рисунок 9 – График производительности, при внутреннем объединении записей

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.

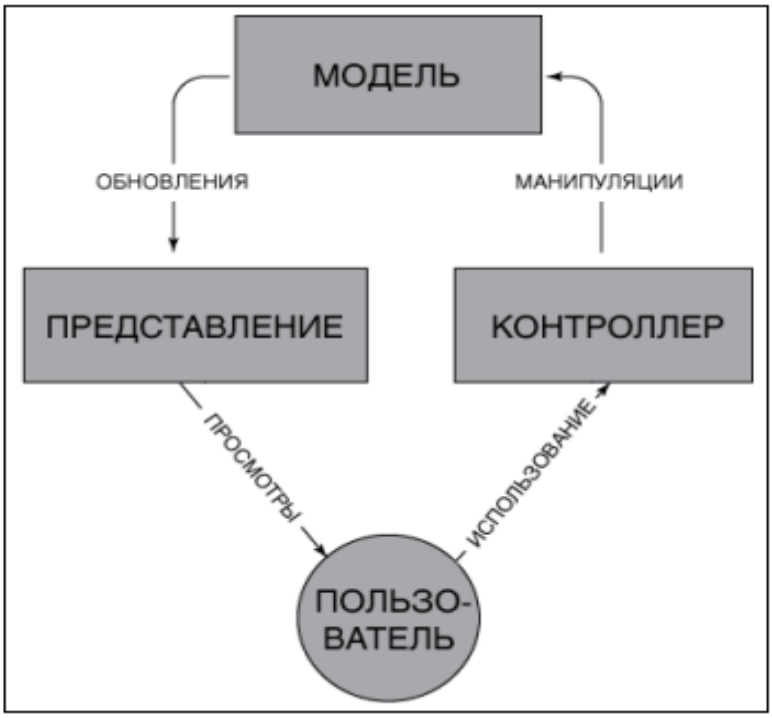

Рисунок 10 – Схема Model-View-Controller

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 сетевая схема системы, представленная на рисунке 11 Диаграмма показывает устройство компьютерной сети разрабатываемой системы. В систему введены маршрутизаторы для создания отдельных подсетей, для обеспечения безопасности и независимости групп элементов сети.

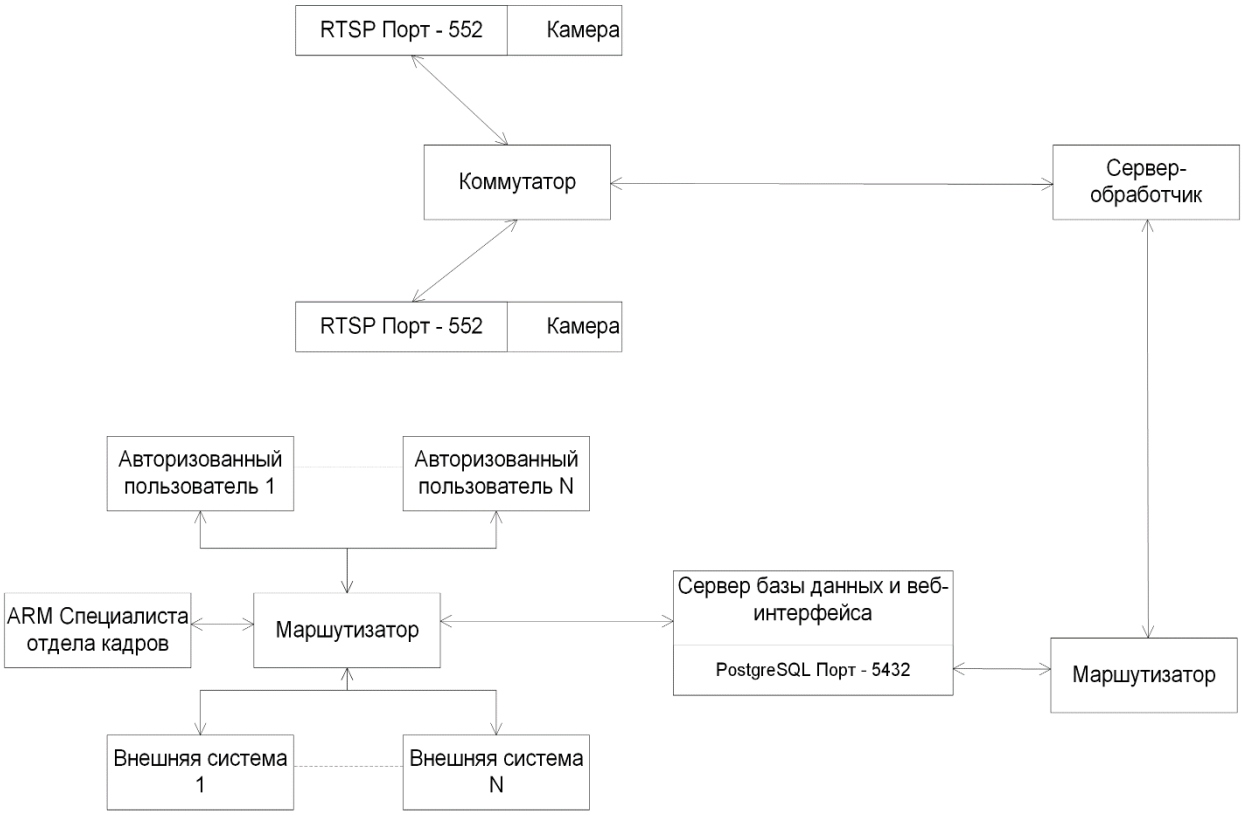

Рисунок 11 – Сетевая схема системы

Исходя из структурной и функциональной схем список оборудования будет состоять из следующего оборудования:

- 56 камер;
- 3 однотипных коммутаторов для объединения камер на каждом этаже;
- 1 сервер базы данных и веб-интерфейса;
- 3 сервер обработчика.

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)».

Коммутаторы, необходимые для объединения камер на каждом этаже должны быть однотипными, должен быть настраиваемым либо управляемы, для настройки работающих портов подключения, иметь количество портов более 22. Исходя из требования выбрана модель – «Utepo UTP-7224E-POE-L2».

Коммутатор, необходимый для объединения серверов системы, должен обладать схожии требования, как и вышеописанные, кроме количества портов. Так же 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 – «D-Link DES-3200-28/C1A».

Сервера-обработчики должны быть однотипными, далее будут изложены требования к одному:

- минимум 24 потоков процессра;

- наличие оперативной памяти типа «DDR4» и объёмом 32 Гб – для обеспечения быстроты работы контейнеров;

- наличие твердотельного накопителя объёмом 500 Гб – для быстроты работы операционной системы.

Исходя из вышеописанных требований была выбрана модель – «MulitOffice 7C137D32S512IMG7».

Для сервера базы данных и веб-интерфейса необоходимо соблюдение следующих условий:

- 8 ядер центрального процессора – для работы двух контейнеров, обрабатывающих данные с камер;

- наличие оперативной памяти типа «DDR4» и объёмом 8 Гб – для обеспечения быстроты работы контейнеров;

- наличие твердотельного накопителя объёмом 125 Гб – для быстроты работы операционной системы «Jet Office 7i10700D8HD05SD12VGALW50».

На каждом из серверов устанавливаются программы «Docker», «Docker-compose», с помощью которых разворачиваются контейнеры с программами.

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 таблицу стилей css. Для развёртывания системы на операционной системе серверов используется программа Docker, позволяющая быстро и безопастно развёртывать программное обеспечение. Разработаны диаграмма сетевой топологии и диаграмма компонентов. Для развёртывания системы было выбрано следующее оборудование:

- 56 «Hikvision DS-2CD2046G2-IU/SL(C)»;

- 3 коммутатора Utepo UTP-7224E-POE-L2

- 1 коммутатор «D-Link DES-3200-28/C1A»;

- 3 сервера-обработчика «MulitOffice 7C137D32S512IMG7»;

- один сервер базы данных и веб-интерфейса «Jet Office 7i10700D8HD05SD12VGALW50».