**«Санкт-Петербургский государственный электротехнический университет**

**«ЛЭТИ» им. В.И.Ульянова (Ленина)»**

**(СПбГЭТУ «ЛЭТИ»)**

|  |  |  |
| --- | --- | --- |
| **Направление** | 27.03.04 – Управление в технических  системах | |
| **Образовательная программа** | Системы и технические средства  автоматизации и управления | |
| **Факультет** | ФЭА | |
| **Кафедра** | САУ | |
| *К защите допустить* |  | |
| Зав. кафедрой |  | Шелудько В.Н. |

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

БАКАЛАВРА

Тема: УПРАВЛЕНИЕ ПРОПУСКНОЙ СИСТЕМОЙ С ПОМОЩЬЮ РАСПОЗНАВАНИЯ ЛИЦ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | *8491* |  |  | Каримжонов Х.Т. |
|  |  | *подпись* |  |  |
| Руководитель | к.т.н., доцент |  |  | Никоза А.В. |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |
| Консультанты | к.т.н., доцент |  |  | Иванов А.Н. |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |
|  |  |  |  | Кузьмина Т.О. |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |

Санкт-Петербург

2022

**ЗАДАНИЕ**

**на выпускную квалификационную работу**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | Утверждаю | | | | | | | |
|  | | Зав. кафедрой САУ | | | | | | | |
|  | | Шелудько В.Н. | | | | | | | |
|  | | «18» марта 2022 г. | | | | | | | |
|  | |  | | | | | | | |
| Студент | Каримжонов Х.Т. | | | | |  | Группа | 8491 | |
| Тема работы**:** Управление пропускной системой с помощью распознавания лиц | | | | | | | | | |
| Место выполнения ВКР: Кафедра САУ | | | | | | | | | |
| Исходные данные (технические требования):  Камера, Ethernet порт, язык программирование Python, библиотеки: face\_recognition, opencv, pygame, imutils, mediapipe | | | | | | | | | |
| Содержание ВКР:  1. Анализ предметной области, распознавание лиц.  2. Сокетное программирование.  3. Модель взаимодействия клиент-сервер.  4. Симулятор устройства исполнитель.  Система предназначена для контроля и управления доступом персонала с помощью которых решается задача контроля и управления посещением охраняемого объекта работниками и посетителями. | | | | | | | | | |
| Перечень отчетных материалов: пояснительная записка, презентация | | | | | | | | | |
| Дополнительные разделы: Безопасность жизнедеятельности | | | | | | | | | |
|  | | | | | | | | | |
| Дата выдачи задания | | | | Дата представления ВКР к защите | | | | | |
| «4» февраль 2022 г. | | | | «4» июня 2022 г. | | | | | |
|  | | | |  | | | | | |
| Студент 8491 | | |  | | Каримжонов Х.Т. | | | |
| Руководитель к.т.н., доцент | | |  | | Никоза А.В. | | | |
| *(Уч. степень, уч. звание)* | | |  | |  | | | |

**календарный план выполнения**

**выпускной квалификационной работы**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой САУ |
|  | Шелудько В.Н. |
|  | «18» марта 2022 г. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | Каримжонов Х.Т. |  | Группа | 8491 |
| Тема работы: Управление пропускной системой с помощью распознавания лиц | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № п/п | Наименование работ | | | Срок выполнения | |
| 1 | Обзор литературы по теме работы | | | 09.02 – 31.02 | |
| 2 | Анализ предметной области | | | 01.03 – 29.03 | |
| 3 | Устройство центрального управления «сервер» | | | 02.04 – 27.04 | |
| 4 | Устройства исполнители «замки» | | | 08.05 – 20.05 | |
| 5 | Безопасность жизнедеятельности | | | 08.05 – 20.05 | |
| 6 | Оформление пояснительной записки | | | 08.05 – 20.05 | |
| 7 | Оформление иллюстративного материала | | | 20.05 – 26.05 | |
| Студент 8491 | |  | Каримжонов Х.Т. | |
| Руководитель к.т.н., доцент | |  | Никоза А.В. | |
| *(Уч. степень, уч. звание)* | |  |  | |

**РЕФЕРАТ**

Пояснительная записка 50 листов, 41 рисунка, 9 источников, 2 приложений.  
СИСТЕМА КОНТРОЛЯ И УПРАВЛЕНИЯ ДОСТУПОМ, РАСПОЗНАВАНИЯ ЛИЦ, НЕЙРОННЫЕ СЕТИ, ВИДЕОФИКСАЦИЯ

Цель работы — разработка системы контроля и управления доступом  
персонала. Объектом разработки является программный комплекс, обеспечивающий обработку данных, поступающих с видеокамеры.

В результате проведенной работы создан проект, удовлетворяющий  
поставленной задаче, а именно, создан программный комплекс, в состав которого вошел веб интерфейс для управления базой данных, алгоритм аутентификации и симулятор устройства исполнителя.

Данное устройство рекомендуется использовать для организации  
ограничения доступа на территорию. Устройство может быть внедрено на  
предприятия, где требуется ограничить круг лиц, имеющих право доступа на  
закрытую территорию

ABSTRACT

The purpose of the work is to develop an access control and management system staff. The object of development is a software package that provides processing of data coming from video cameras.

As a result of the work carried out, a project was created that sets up task, namely, the created software package, which includes a web interface for managing the database, an authentication algorithm and a simulator of the executor's device.

This device is recommended for organization the likelihood of a problem. The device can be embedded enterprises where an exclusive circle of people is required, closed work

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 7](#_Toc105146549)

[1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 9](#_Toc105146550)

[1.1 СКУД персонала 9](#_Toc105146551)

[1.2 Основное предназначение СКУД и типовые сферы применения 9](#_Toc105146552)

[1.3 СКУД с помощью распознавания лиц 11](#_Toc105146553)

[1.4 Принцип работы распознавания лиц 12](#_Toc105146554)

[1.5 Модель взаимодействия клиент-сервер 19](#_Toc105146555)

[1.6 Постановка задач 21](#_Toc105146556)

[2 УСТРОЙСТВА ИСПОЛЬНИТЕЛИ «ЗАМКИ» 23](#_Toc105146557)

[2.1 Принцип работы «Замка» 23](#_Toc105146558)

[2.2 Передача видео потока и получение ответа 24](#_Toc105146559)

[2.3 Симулятор устройства исполнителя 25](#_Toc105146560)

[3 УСТРОЙСТВО ЦЕНТРАЛЬНОГО УПРАВЛЕНИЯ «СЕРВЕР» 29](#_Toc105146561)

[3.1 Принцип работы «Сервера» 29](#_Toc105146562)

[3.2 Алгоритм аутентификации 31](#_Toc105146563)

[3.3 Веб интерфейс для управление данных персонала 33](#_Toc105146564)

[3.4 Графический интерфейс для отслеживания в реальном времени 38](#_Toc105146565)

[4 БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ 41](#_Toc105146566)

[4.1 Общие положения 41](#_Toc105146567)

[4.2 Аспект эргономики программного обеспечения 41](#_Toc105146568)

[4.3 Соответствие разработанного ПО требованиям ГОСТ Р ИСО 9241 44](#_Toc105146569)

[4.4 Соответствие использованного ПО требованиям ГОСТ Р ИСО 9241 46](#_Toc105146570)

[ЗАКЛЮЧЕНИЕ 49](#_Toc105146571)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 50](#_Toc105146572)

[ПРИЛОЖЕНИЕ А 51](#_Toc105146573)

[ПРИЛОЖЕНИЕ Б 55](#_Toc105146574)

# ВВЕДЕНИЕ

Система контроля и управления доступом (СКУД) – это совокупность программных и технических средств, а также организационно-методических мероприятий, с помощью которых решается задача контроля и управления посещением охраняемого объекта.

СКУД может решать такие задачи, как оперативный контроль местонахождения персонала и их автотранспорта, время нахождения персонала на объекте. С помощью программных средств реализуется функция расчёта отработанного сотрудниками времени. Зачастую на предприятиях СКУД интегрируется с системой охранно-пожарной сигнализации для комплексного решения задач безопасности. Тем самым можно обеспечить реакции охранной системы на попытки несанкционированного доступа, взлома дверей и т.п. Возможность автоматической постановки/снятия с охраны помещений по факту прохода в зону доступа сотрудника; предоставление свободного выхода/выезда в случае возникновения пожара.

При реализации конкретных СКУД используют различные способы и реализующие их устройства для идентификации и аутентификации личности. Следует отметить, что СКУД являются одним из наиболее развитых сегментов рынка безопасности как в России, так и за рубежом. По данным ряда экспертов ежегодный прирост рынка СКУД составляет более 25 %. Число специалистов, работающих в сфере технических систем безопасности, превысило 500 тыс. человек. Это связано с тем, что, во-первых, постепенно повышается информированность рынка и как следствие востребованность новых возможностей, функций и сервисов, которые не могли быть реализованы в рамках более старых классических сегментов рынка систем безопасности. Во-вторых, на повышение рыночной динамики значительно влияют такие факторы, как увеличение риска террористических угроз, рост общего уровня культуры потребителей.

СКУД на данный момент времени большинством пользователей воспринимается как важная составная часть системы безопасности предприятия. Значительную роль в достижении такого результата сыграла и продолжает играть именно информированность конечных потребителей. С ростом информированности закономерно повышается уровень требований к СКУД, так, например, на объектах, где требуется обеспечить повышенный уровень безопасности (аэропорты, ядерные объекты, промышленные предприятия), стали использоваться системы биометрической идентификации (по отпечатку пальца, форме ладони, радужной оболочке, чертам лица), в том числе и многофакторной по комбинации биометрических признаков и пароля или карты доступа. В других организациях, предприятиях, учебных заведениях и т.д. используются турникеты с бесконтактными картами доступа; шлагбаумы с пультом доступа или видеофиксацией.

В рамках данной дипломной работы было принята разрабатывать программный комплекс, который продемонстрирует работа СКУД и предстоит решить следующий задач:

* Передача видео потока и получение ответа
* Симулятор устройство исполнителя
* Получения данные от устройства исполнители
* Алгоритм аутентификации
* Сохранение потокового информации
* Веб интерфейс для управление данных персонала
* Отслеживание персоналом в реальном времени

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## **1.1 СКУД персонала**

Система представляет аппаратно-программный комплекс, с помощью  
которого решается задача организации автоматического или  
автоматизированного проход на подконтрольную территорию и учета  
перемещения персонала. Основная задача — управление доступом на  
заданную территорию (кого пускать и на какую территорию).  
Система контроля и управления доступом персонала может входить  
в общую систему СКУД или действовать независимо. Основным назначением системы является учет и контроль доступа персонала, мониторинг и перемещения по территории.

## **1.2 Основное предназначение СКУД и типовые сферы применения**

Основное предназначение СКУД - гибкая настройка уровней и порядка доступа персонала, посетителей и автотранспорта на территорию объекта с различными вариантами идентификации. Система контроля и управления доступом позволяет создавать различные правила доступа, автоматизировать процесс выдачи временных пропусков, внедрять интеллектуальное управление исполнительными механизмами, оформлять заявки на гостевые пропуска, интегрироваться с системами регистрации, учета и контроля посетителей. Автоматизированная система управления пропускной системы предназначена для организации контроля прохода персонала через точки контроля с организацией требуемого алгоритма аутентификации и сбора данных о персонале [[1]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ).

При работе с персоналами КТП в число данных могут входить такие как:

- кодировка лиц персонала

- направление движения (вход/выход),

- полномочия персонала (пропуск, принадлежность группе, согласно  
заранее внесенных данных),

- дата/время прохождения точки контроля,

Типовые применение такой системы является:

• Организация автоматического проход на охраняемую территорию.

• Автоматизация проход в жилых помещениях.

• Организация проход по территорию бизнес-центров.

• Учет перемещения персонала на цехах, в школах и других общественных мест.

• Разграничение доступа и учет перемещения по территории.

Основной функционал системы СКУД:

1. Контроль перемещения по территорию предприятия или других  
организаций:

• Идентификация персонала средств по распознанному лицу.

• СКУД проводит автоматически проверку разрешения прохода  
персонала на территорию предприятия. Устанавливает временной интервал на разрешение проезда и выезда;

• СКУД не допускает пребывания на территорию чужого  
персонала без регистрации у служб охраны предприятия;

2. СКУД контролирует и самостоятельно архивирует в журнале событий все перемещения персонала

3. Удобное администрирование через веб-интерфейсы

4. Управление замками, воротами и другими преградами.

5. Взаимодействие с системой СКУД и другими системами, в том  
числе учета рабочего времени, охранной и пожарной сигнализации,  
видеонаблюдения, охраны периметра и др.

## **1.3 СКУД с помощью распознавания лиц**

Технологии машинного зрения и распознавания лиц развивались очень активно с середины прошлого века. Но только сейчас стали по-настоящему хорошо работать. На это есть три причины:

1. Появились действительно мощные компьютеры, способные справиться с такой задачей. За это спасибо закону Мура.

2. Появились базы данных с нашими с вами фотографиями. За что спасибо социальным сетям.

3. Ну и конечно, произошел прорыв в области нейросетей.

Все эти события позволили создать практически идеальные алгоритмы распознавания лиц.

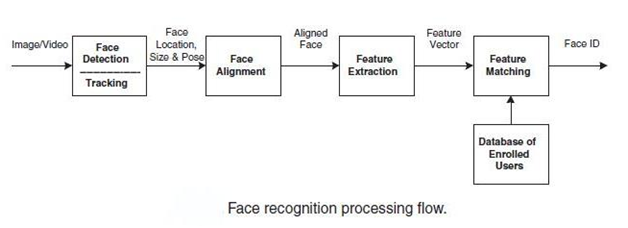


Рисунок 1.1 – Технология распознавание лиц.

Современный технический прогресс затронул и сферу безопасности, а потому то, что казалось фантастикой еще несколько десятилетий назад, сейчас с успехом воплощается в жизнь благодаря новым технологиям. Пользователем стал доступен ряд новейших и модернизированных охранных систем, включая модули с функцией распознавания лиц. Основным достоинством такой системы является бесконтактный способ идентификации, при котором не требуется соприкосновение считывателя и идентификатора (карты, брелока, ключа, метки и т.д.). Полученное лицо персонала кодируются и отправляется в центральную управление (сервер) где в автоматическом режиме  
принимается решение о разрешении доступа [[2]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ).

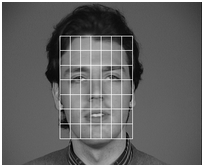
## **1.4 Принцип работы распознавания лиц**

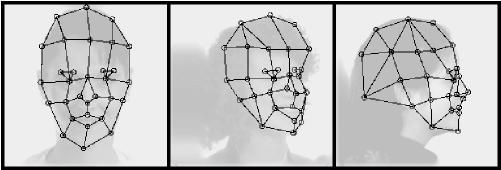
Несмотря на большое разнообразие представленных алгоритмов, можно выделить общую структуру процесса распознавания лиц:

  
Рисунок 1.2 – Общий процесс обработки изображения лица при распознавании

На первом этапе производится детектирование и локализация лица на изображении. На этапе распознавания производится выравнивание изображения лица (геометрическое и яркостное), вычисление признаков и непосредственно распознавание – сравнение вычисленных признаков с заложенными в базу данных эталонами. Основным отличием всех представленных алгоритмов будет вычисление признаков и сравнение их совокупностей между собой [[3]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ).

**Метод гибкого сравнения на графах (Elastic graph matching).** Суть метода сводится к эластичному сопоставлению графов, описывающих изображения лиц. Лица представлены в виде графов со взвешенными вершинами и ребрами. На этапе распознавания один из графов – эталонный – остается неизменным, в то время как другой деформируется с целью наилучшей подгонки к первому. В подобных системах распознавания графы могут представлять собой как прямоугольную решетку, так и структуру, образованную характерными (антропометрическими) точками лица.

а)   
   
б)   
Рисунок 1.3 – Пример структуры графа для распознавания лиц (а - регулярная решетка, б - граф на основе антропометрических точек лица)

В вершинах графа вычисляются значения признаков, чаще всего используют комплексные значения фильтров Габора или их упорядоченных наборов – Габоровских вейвлет (строи Габора), которые вычисляются в некоторой локальной области вершины графа локально путем свертки значений яркости пикселей с фильтрами Габора.

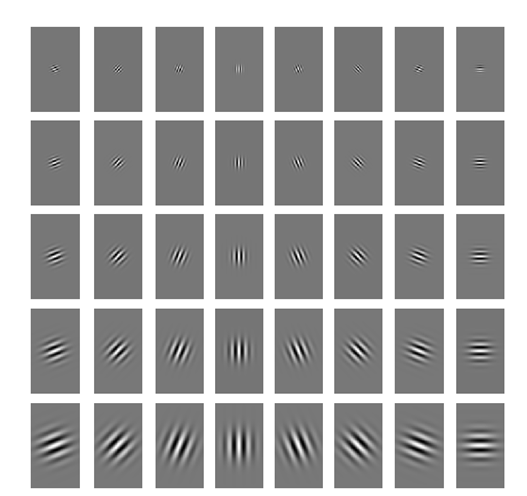
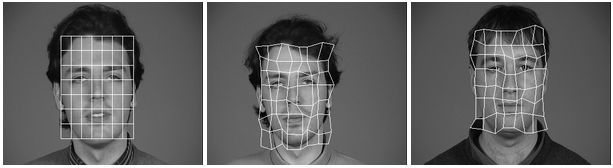
  
Рисунок 1.4 – Набор (банк, jet) фильтров Габора

  
Рисунок 1.5 – Пример свертки изображения лица с двумя фильтрами Габора

Ребра графа взвешиваются расстояниями между смежными вершинами. Различие (расстояние, дискриминационная характеристика) между двумя графами вычисляется при помощи некоторой ценовой функции деформации, учитывающей как различие между значениями признаков, вычисленными в вершинах, так и степень деформации ребер графа.

Деформация графа происходит путем смещения каждой из его вершин на некоторое расстояние в определённых направлениях относительно ее исходного местоположения и выбора такой ее позиции, при которой разница между значениями признаков (откликов фильтров Габора) в вершине деформируемого графа и соответствующей ей вершине эталонного графа будет минимальной. Данная операция выполняется поочередно для всех вершин графа до тех пор, пока не будет достигнуто наименьшее суммарное различие между признаками деформируемого и эталонного графов. Значение ценовой функции деформации при таком положении деформируемого графа и будет являться мерой различия между входным изображением лица и эталонным графом. Данная «релаксационная» процедура деформации должна выполняться для всех эталонных лиц, заложенных в базу данных системы. Результат распознавания системы – эталон с наилучшим значением ценовой функции деформации.

  
Рисунок 1.6 – Пример деформации графа в виде регулярной решетки

В отдельных публикациях указывается 95-97%-ая эффективность распознавания даже при наличии различных эмоциональных выражениях и изменении ракурса лица до 15 градусов. Однако разработчики систем эластичного сравнения на графах ссылаются на высокую вычислительную стоимость данного подхода. Например, для сравнения входного изображения лица с 87 эталонными тратилось приблизительно 25 секунд при работе на параллельной ЭВМ с 23 транспьютерами.

Недостатки: высокая вычислительная сложность процедуры распознавания. Низкая технологичность при запоминании новых эталонов. Линейная зависимость времени работы от размера базы данных лиц [[3]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ).

**Нейронные сети. Обнаружение.** В первую очередь, для того, чтобы лицо распознать, надо его сначала обнаружить. Задача на самом деле не тривиальная. Для этого мы бы могли использовать натренированные нейросети, но это слишком долго, дорого и ресурсоемко. Поэтому для обнаружения лица используется очень простой метод Виолы — Джонса, разработанный еще в 2001 году.

Этот алгоритм просто сканирует изображение при помощи таких прямоугольников (см. рис. 1.8), они называются примитивами Хаара:

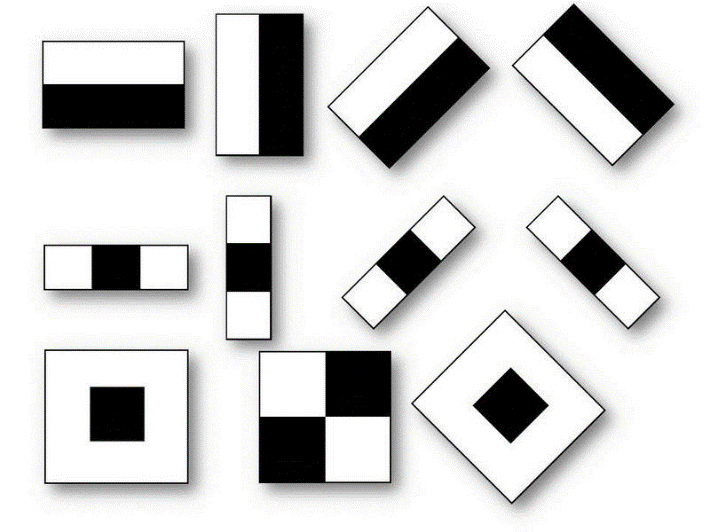


Рисунок 1.7 – Примитивы Хаара

Задача этих объектов — находить более светлые и темные области на изображении, характерных конкретно для человеческих лиц.

Например, если усреднить значения яркости область глаз будет темнее щек или лба, а переносица будет светлее бровей. (см. рис. 1.9)

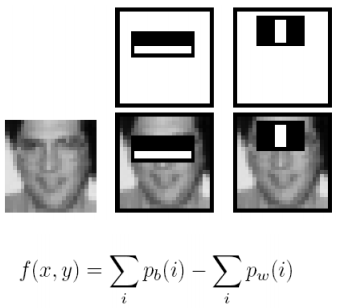


Рисунок 1.8 – Пример использование примитивы Хаара

В общем таких характерных признаков много и естественно не только у человеческих лиц могут быть подобные паттерны. Поэтому алгоритм работает в несколько этапов:

Сначала находится первый признак, система понимает: «В этой области может быть лицо». Тогда она начинает там же искать второй признак, а потом третий. И если в одной области найдено 3 признака, уже можно уверенно сказать — да, это лицо! После чего система получает область изображения, в котором есть только лицо.

**Антропометрические точки.** Получив область для анализа, дальше в дело вступает главный секрет каждой системы распознавания — биометрический алгоритм.

Он расставляет на лице антропометрические точки, по которым впоследствии и будут вычисляться индивидуальные характеристики человека: разрез глаз, форма носа, подбородка, расстояние между ними и прочее. Таких признаков может быть много, вплоть до нескольких тысяч. Но в целом, таких точек должно быть 468. (см. рис. 1.10)

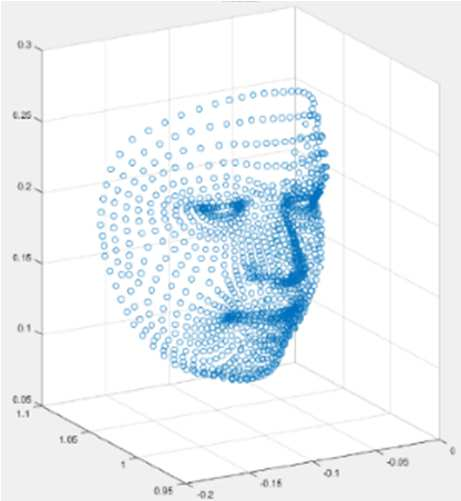


Рисунок 1.9 – Сетка лица

**Исправление искажений.** А дальше начинается настоящая магия. В идеале нам нужно лицо, которое смотрит анфас, то есть прямо в камеру. Но такая удача бывает редко, особенно если речь идет о распознавании человека в толпе.

Поэтому система производит дополнительное преобразование изображения: устранятся поворот и наклон головы. А также проводится 3D-реконструкция лица из 2D-изображения. Таким образом, даже если человек на изображении смотрел вбок, мы всё равно можем получить четкий фронтальный снимок, что существенно повышает качество распознавания. (см. рис. 1.11)

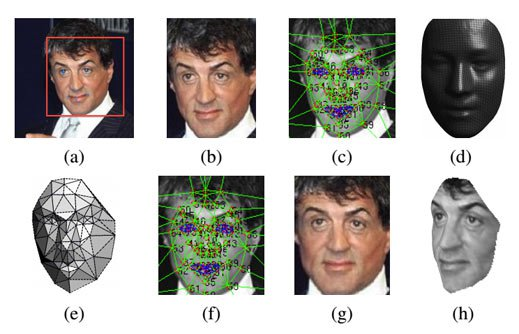


Рисунок 1.10 – Пример исправление искажение

**Вектор лица.** Ну а дальше происходит самое главное. В бой вступает нейросеть, которая присваивает каждому лицу вектор признаков.

По сути, это просто какое-то число, которое складывается из суммы характеристик лица: расстояний между опорными точками, текстуры определенных областей на лице и прочее. Таких характеристик может быть множество. Основное правило: они должны описывать лицо независимо от посторонних факторов: макияжа, прически, возрастных изменений.

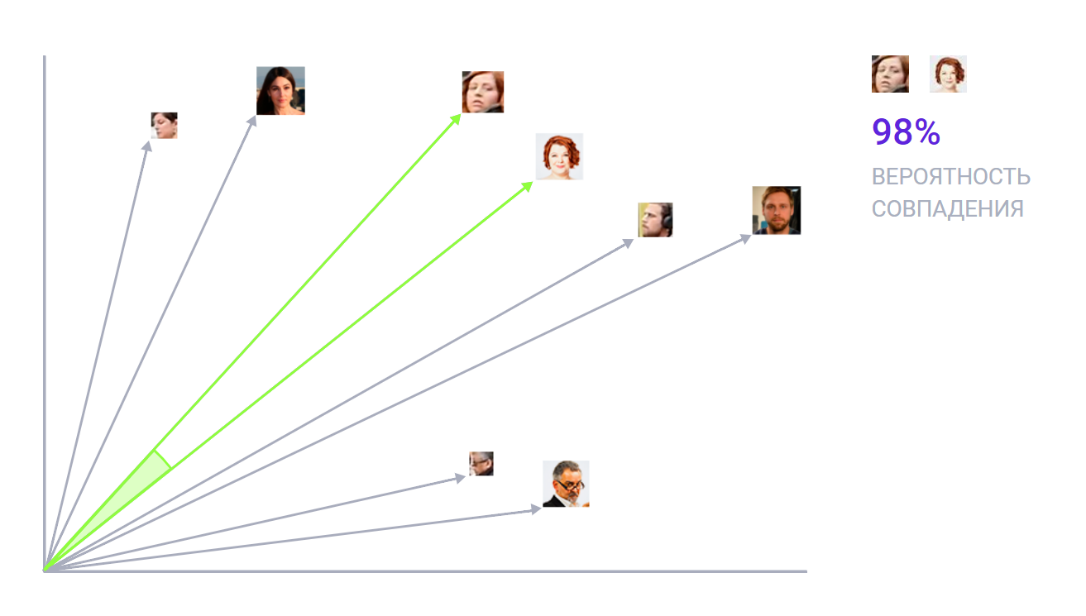


Рисунок 1.11 – Сравнение с базой данных

**Идентификация.** Ну а дальше остаётся сравнить полученный вектор с базой других векторов. И готово. Система вас идентифицировала [[3]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ).

**1.5 Модель взаимодействия клиент-сервер**

М**одель взаимодействия клиент-сервер - это** разделение функционала и вычислительной нагрузки между клиентскими и серверными приложениями.

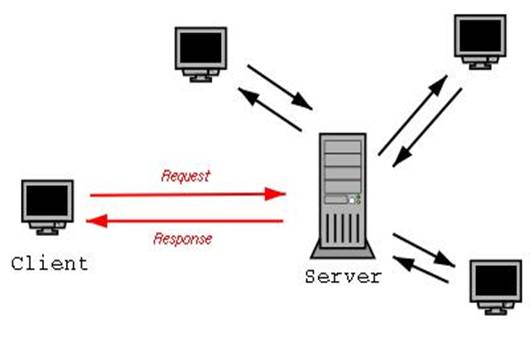


Рисунок 1.12 – Модель взаимодействия клиент-сервер.

Обмен информацией между клиентом и сервером происходит благодаря сетевым протоколам в интернете. Каждой услуге соответствует определенный протокол, в нашем случае мы будем использовать протокол IP(INET) и сокетное программирование.

Сокеты - название программного интерфейса для обеспечения обмена данными между процессами. Сокет - абстрактный объект, представляющий конечную точку соединения.

Каждый процесс может создать слушающий сокет (серверный сокет) и привязать его к какому-нибудь порту операционной системы. Слушающий процесс обычно находится в цикле ожидания, то есть просыпается при появлении нового соединения. При этом сохраняется возможность проверить наличие соединений на данный момент, установить тайм-аут для операции и многое другое.

Каждый сокет имеет свой адрес. Сокеты типа INET доступны из сети и требуют выделения номера порта.

Обычно клиент явно подсоединяется к слушателю, после чего любое чтение или запись через его файловый дескриптор будут передавать данные между ним и сервером [[8]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ).

**Преимуществом модели взаимодействия клиент-сервер является то, что программный код клиентского приложения и серверного разделен**. Если мы говорим про локальные компьютерные сети, то к преимуществам архитектуры клиент-сервер можно отнести пониженные требования к машинам клиентов, так как большая часть вычислительных операций будет производиться на сервере, а также архитектура клиент-сервер довольно гибкая и позволяет администратору сделать локальную сеть более защищенной.

К недостаткам модели взаимодействия клиент-сервер можно отнести то, что стоимость серверного оборудования значительно выше клиентского. Сервер должен обслуживать специально обученный и подготовленный человек. Если в локальной сети ложится сервер, то и клиенты не смогут работать

В качестве заключения стоит явно акцентировать внимание на том, что **архитектура клиент-сервер не делит машины на только клиент или только сервер**, а скорее позволяет распределить нагрузку и разделить функционал между клиентской частью и серверной.

**1.6 Постановка задач**

Процесс аутентификации имеет 2 подсистемы, которые представлены на рисунке 1.14.

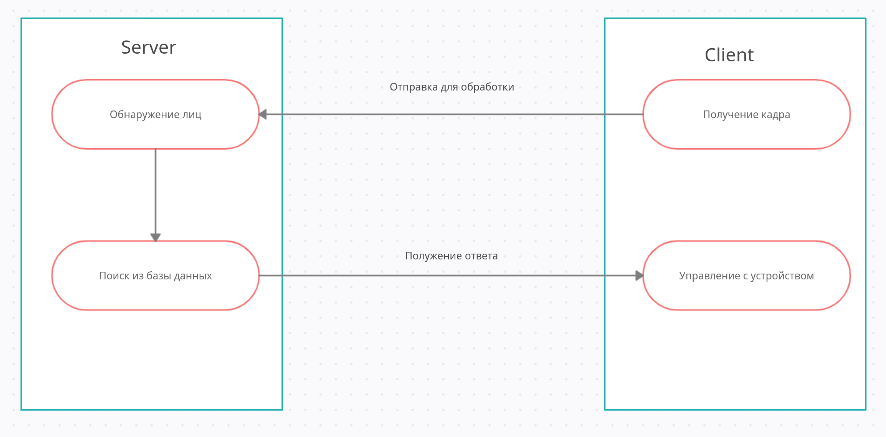


Рисунок 1.13 – Схема аутентификации

Подсистема сервер выполняет роль “мозга”. Она получает кадр, обрабатывает и обнаруживает лиц и сравнивает с базой данных. Если этому человеку разрешено входить, отправляет разрешение, иначе наоборот. Там еще находится управление базой данных, который реализован в языке программирования Python. Сервер будет рассчитан обслуживать много клиентов (замков).

Подсистема Клиент представляет собой замок, имеющий камеры, соединенный в локальный хост. Этот замок имеет микросхему, которую управляет этим замком и камерой.

В рамках данной дипломной работы предстоит решит следующий задач:

* Передача видео потока и получение ответа
* Симулятор устройство исполнителя
* Получения данные от устройства исполнители
* Алгоритм аутентификации
* Сохранение потокового информации
* Веб интерфейс для управление данных персонала
* Отслеживание персоналом в реальном времени

**Выводы по разделу.** В данном разделе рассказано принцип работы и основной функционал данного СКУД. Также в этом разделе было представлено возможные решеные для реализации системы управление такого типа.

# 2 УСТРОЙСТВА ИСПОЛЬНИТЕЛИ «ЗАМКИ»

## **2.1 Принцип работы «Замка»**

Замок управляющей системой пропусков представляет собой 4 компонента:

- микросхема, имеющая Ethernet порт для подключения локальному хосту;

- камера для получения кадра;

- датчик для получения состояния двери (открытый или закрытый);

- язычок для блокировки дверей;

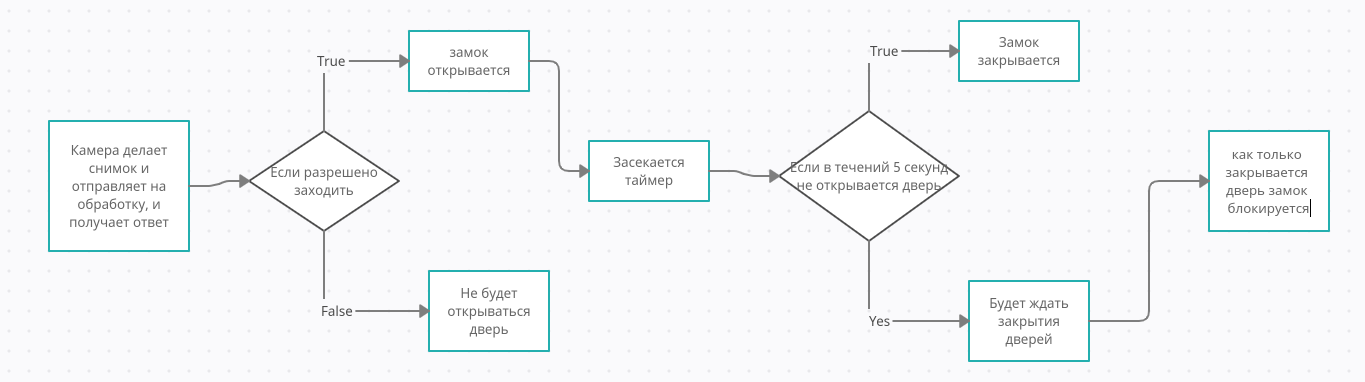


Рисунок 2.1 – Блок схема управления замком

Принцип работы замка предоставляет собой следующие:

Камера захватывает кадр, отправляет его на обработку и получает ответ, который представляет собой булево значение. Если значение не истинно, дверь не будет открываться, иначе язычок притягивается и замок будет открыт в течении 5 секунд. После истечения времени замок будет закрыт. Если человек откроет дверь в течении 5 секунд, датчик срабатывается и толкатель язычок будет ждать до тех пор, пока дверь не закроется. Когда дверь закрывается датчик срабатывает и закрывается замок. И этот цикл будет работать непрерывно.

## **2.2 Передача видео потока и получение ответа**

Передача видео потока в первый взгляд простой, но в ходе выполнения дипломной работы была выбрана низкоуровневое сетевое программирование (сокетное программирование). У этого выбора есть плюсы в виде производительность и безопасность, и есть минусы, все надо написать самому с простого отправка слово “Hello world” до отправки кадра. Для того еще улучшить отзывчивость системы была использована многопоточное программирование.

Сервер (центральное управление) постоянно ожидает подключение по определенному порту новых устройств (по порту 5000 в нашем случае). При подключение новых устройств сервер открывает дополнительный порт для передачи видео кадров, а подключенную соединению использует для отправки решение (открыт дверь или нет). Таким образом мы создаем 2 независимы передача данных и основной достоинство такой системы не надо имеет большой производительности в устройствах все для них делает сервер и путем такая поступка уменьшается себестоимость устройству.

Кадр - это матрица из трех цифр (RGB – красный, зеленый, синий) и будет имеет вид (ширина, высота, 3). При отправке кадра нельзя отправит матрицу или проста цифру. Надо эти данные преобразовать в байт. Есть один примечание что максимум то что можно отправит в одном отправке это 4096 байт если отправит больше при распаковке этих данных будет потеря пакета данных. Что бы устранит такое поведение было написано алгоритм отправке следующем образом.

Например, кадр имеет вид шириной 640 пикселей и высотой 480. При каждой отправке будет отправлено (1, 640, 3) част изображение (кадр 1 пиксел высотой) и в сервере будет обратно это собираться. Для того что бы сервер понимал окончание кадра отправляется байтового слова b’stop’, который будет означать окончание кадра.

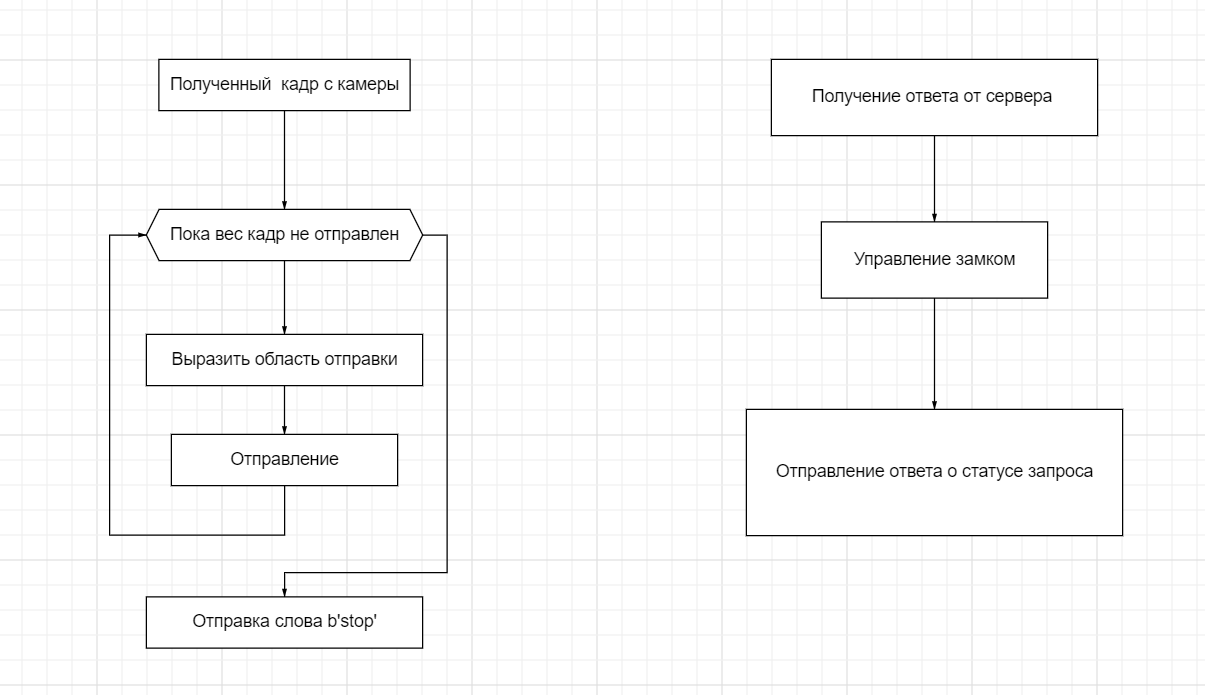


Рисунок 2.2 – Блок схема отправки данных.

В это время другой поток будет делать следующие. Получить ответ с сервера о будущем состоянии замка и управлять замком. После успешного выполнения управления, отправит отчет о состоянии.

## **2.3 Симулятор устройства исполнителя**

В ходе выполнения этой работы было принято не собрать «живой» модель замка, а лишь симулятор, который будет показывать состояние замка (доступ открыт или закрыт), состояние двери (открыто или закрыто) и управления дверей (открыт дверь или закрыт). Для этой цели выбрали библиотека pygame. Основное преимущество этой библиотеки высокая производительность и непрерывно обновления экрана, который нам на руку для просмотра кадров в реальном времени.

С помощью разработанного симулятора можно увидит следующую:

1. Устройство делает снимок и получает ответ от сервера. Этому персонала доступ разрешен и допускается к территорию но персонал пока не заходит (рис. 2.3)

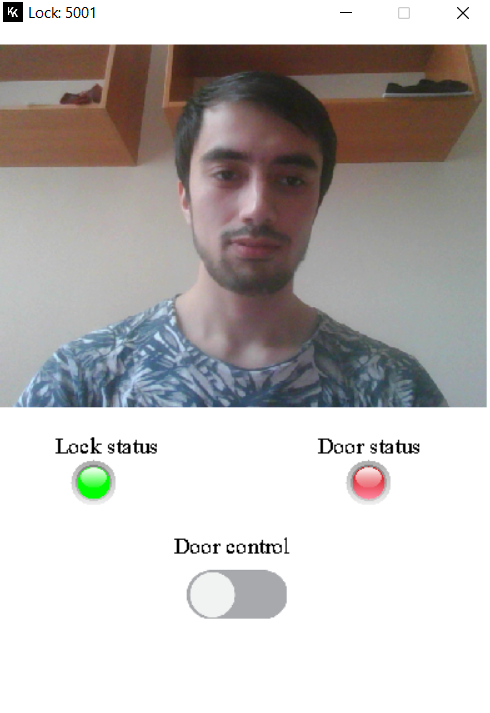


Рисунок 2.3 – Снимок симулятора.

1. Как только персонал открывает дверь (в нашем случи нажимается кнопка Door control) загорается свет Door status означающий о состояние двери (рис. 2.4)

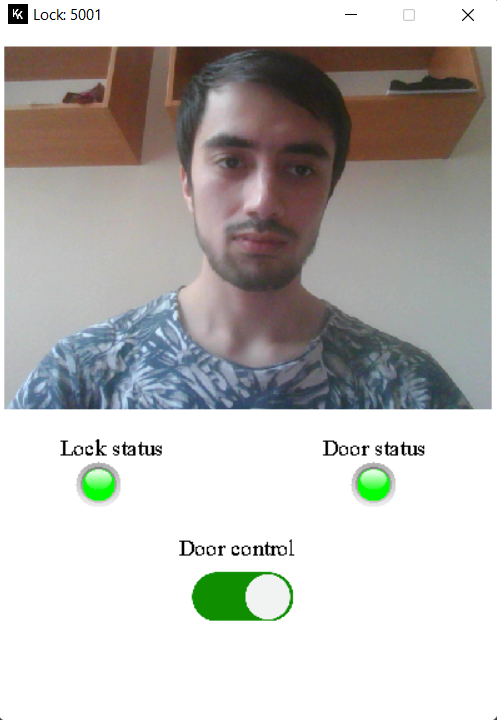


Рисунок 2.4 – Снимок симулятора

1. Пока персонал не закроет дверь (с помощью кнопку Door control) замок не будет закрываться (рис. 2.5)

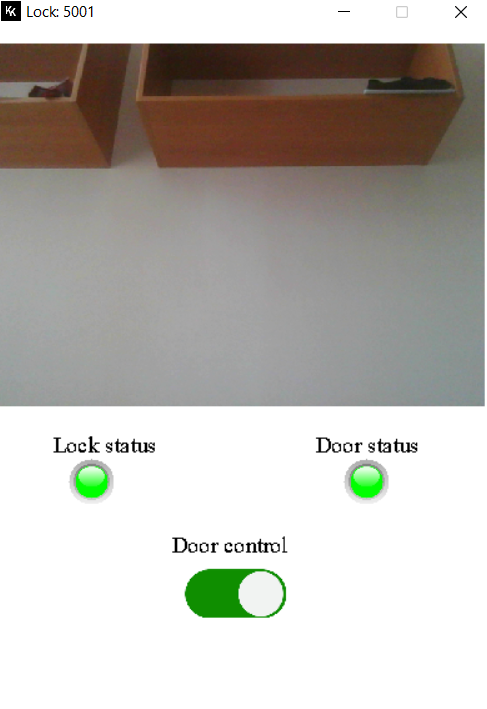


Рисунок 2.5 – Снимок симулятора

1. Как только персонал закроет дверь замок тоже закроется. (рис. 2.6)

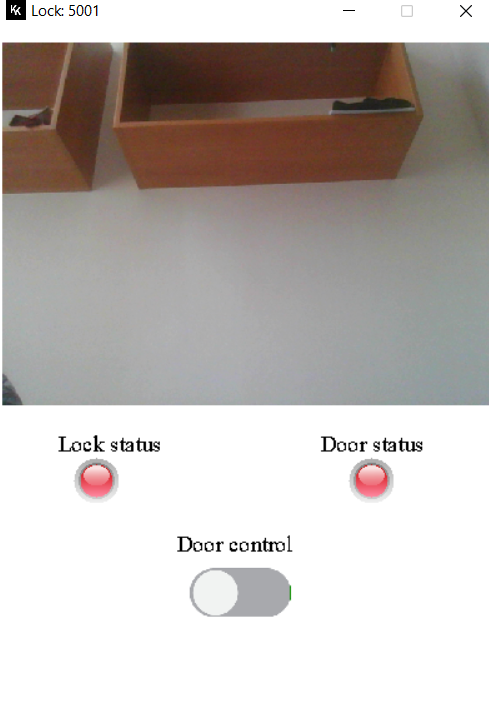


Рисунок 2.6 – Снимок симулятора

Таким образом будет функционировать исполнительные устройств.

**Выводы по разделу.** В данном раздели была разработана симулятор для исполнительных устройств и алгоритм передачи видео потока в реальном времени.

3 УСТРОЙСТВО ЦЕНТРАЛЬНОГО УПРАВЛЕНИЯ «СЕРВЕР»

**3.1 Принцип работы «Сервера»**

Персональный компьютер предназначен для программирования СКУД, получения информации о пользователях системы, дате и времени проезда пользователей через контрольные устройства, срабатывании средств охранно-пожарной сигнализации, видеоконтроля, попыток несанкционированного прохода, аварийных ситуаций.

Для работы в СКУД может использоваться любой персональный IBM  
совместимый компьютер. Наряду с работой в составе СКУД он может  
выполнять и другие функция, т.к. компьютер нужен в основном лишь для  
программирования системы и получения отчетов о работе системы.  
Персональный компьютер, используя специально разработанное для  
охраняемого объекта программное обеспечение (желательно  
русифицированное), осуществляет общее управление и программирование  
СКУД, собирает информацию с контроллеров, создает общий банк данных, формирует различные отчеты и сводки. Русифицированное программное  
обеспечение под MSDOS, Linux или Windows позволяет осуществлять автоматическую запись данных по всем операциям. В любой момент можно  
запросить разнообразные сведения, например, о местонахождении сотрудников и посетителей. Текущее состояние СКУД отображается в удобной графической форме. В компьютер вводится план охраняемого объекта, на котором стандартными значками указываются считыватели, замки, технические  
средства охранно-пожарной сигнализации, видеоконтроля и т.п. На плане  
система автоматически в реальном масштабе времени показывает состояние  
всех нанесенных объектов контроля - открыт или закрыт замок. Таким образом, в любой момент времени можно быстро оценить ситуацию и в случае внештатной ситуации оперативно и эффективно принять меры предосторожности

Программное обеспечение СКУД представляет собой 3 микро-сервиса:

- Data Executer (Передача и получение данных и хранение этих данных, сохранение видео и внесение данных о перемещении персонала).

- Веб интерфейс (для управления данных персонала, cоздавать и изменить карту персонала, загружать фотографию персонала и т.д.).

- Графический интерфейс (для отслеживания камер и управление устройством исполнители)

Выше была упомянута, что используется многопоточное программирование и для того, чтобы у всех потоках были доступы к полученным данным из устройства исполнителей, данные будут хранится во временном хранилище и эти потоки будет иметь следующий сценарий:

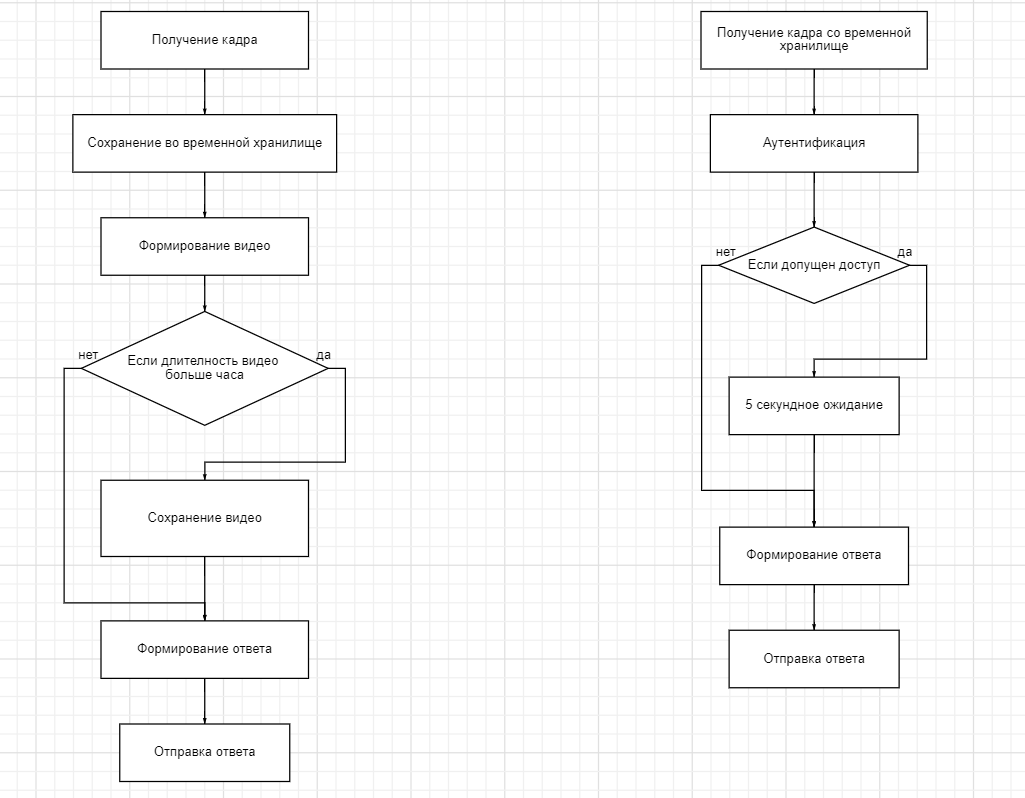


Рисунок 3.1 – Блок схема работы сервера

* Первый поток получает кадр и помещает его во временном хранилище и из получаемых кадров будет собраться видео. Это видео будет сохранится каждый час с названием ГГГГ\_ММ\_ДД\_\_\_ЧЧ\_ММ\_СС.mp4.
* Второй поток берет кадр из временного хранилище и начинает его обрабатывать. При обработке кадра проходит 3 уровни аутентификации. После аутентификации, если персоналу допущен доступ, тогда отправляется положительный ответ и 5 секунд дается время для прохода по устройствам исполнителей. В это время сервер не будет обрабатывать кадры с этого устройства (это сделано для того чтобы снять нагрузку с сервера). А если персоналу не допущен доступ или в кадре не обнаружено лицо человека тогда отправляется отрицательный ответ.
  1. **Алгоритм аутентификации**

Процесс аутентификации состоит из трех частей и будет выполнятся по очереди. Если один из стадии аутентификации вернет отрицательный характер, то сразу объявляется о не допуске (например, если человек находится не в пределе допустимого расстояния, то процесс распознавания этого человека не выполняется).

**Определение дистанции между человеком и камерой.** Для своего временного использования вычислительной мощности сервера, надо задавать порог, который при прохождении этого порога должен открывать замок. Для этого воспользуется фундаментальные знания по оптике.

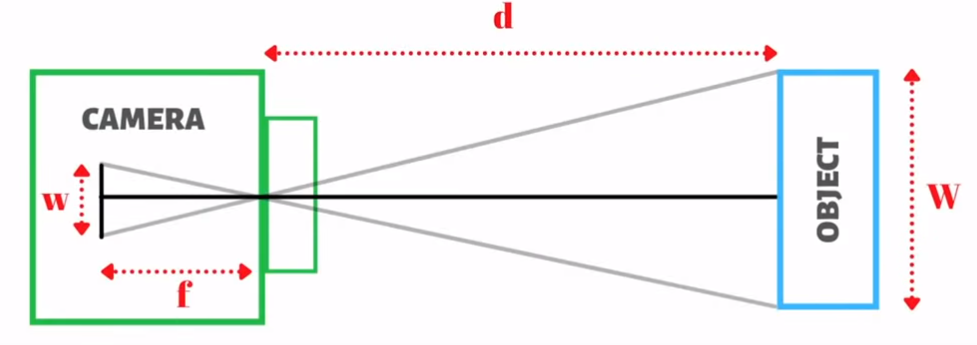


Рисунок 3.2 – Фокальное расстояние

где W - ширина объекта, d - расстояние между камерой и объектом, w - ширина объекта в изображении и f - фокальное расстояние.

В качестве ширины объекта принимается расстояние между глаз, который составляет для мужчин 6.4см, а для женщин 6.2 см. В данной работе принимаем среднее значение W = 6.3 см.

Чтобы найти фокальное расстояние, экспериментальном путем подходя к камере на определенное расстояние, рассчитывается фокальное расстояние.

С помощью этого значения можно будет найти расстояние до человека по этой формуле:

Если человек находится в пределе этого расстояния, тогда будет отрабатываться вторая стадия аутентификации, иначе отправляется отрицательный ответ.

**Распознавание персонала.** Когда речь идет о нейронных сетях и data science, Python славится своими эффективными и достаточно удобными библиотеками.

Библиотека Face Recognition - это Open Source продукт разработано программистом Адам Геитгей и популярно используются для распознавания лиц и нахождение их координаты в высоком уровне. Создан с использованием передовой технологии распознавания лиц dlib, построен с глубоким обучением нейронных сетей. Модель имеет точность 99,38% на Тест Labeled Faces in the Wild [[6]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ).

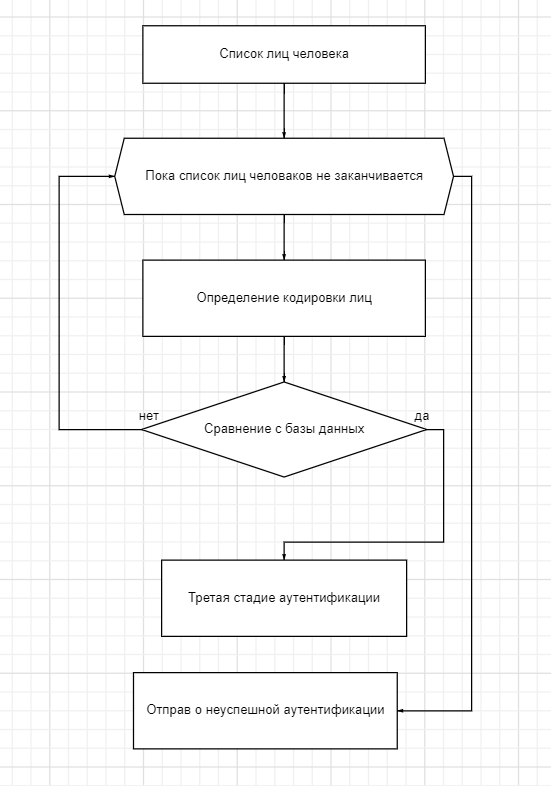


Рисунок 3.3 – Блок схема распознавание лиц

Во второй стадии аутентификации, имея координаты лица человека используя библиотеки face recognition вычисляется кодировки лица человека и будет сравниваться с базы данных, если есть схожесть. Нужно отметить в одном кадре может находится два и более человека. В этом случае, если есть хот один схожесть кодировки человека в базе данных, то вторая стадия аутентификации объявляется успешном, иначе отправляет отрицательный ответ.

После успешного выполнение второй стадии, проверяется есть ли у этого персонала полномочие на эту территорию.

* 1. **Веб интерфейс для управление данных персонала**

Еще одно из преимуществ такого СКУД является удобный веб интерфейс для создания, обновления и удаления карточек персонала при необходимости. Для таких дел особенно Python хорош, который имеет разные библиотеки, который с этом справляется хорошо. Например, Django, Flask, Fast Api и другие. В рамках данной работы было выбрано фреймворк Django.

Django — это высокоуровневая веб-инфраструктура Python, которая способствует быстрой разработке и чистому, прагматичному дизайну. Он берет на себя большую часть хлопот веб-разработки, поэтому можно сосредоточиться на написании своего приложения, не изобретая велосипед. Это бесплатно и с открытым исходным кодом [[7]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ).

Для работы с базой данных Django использует собственный ORM, в котором модель данных описывается классами Python и по ней генерируется схема базы данных.

ORM — технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных».

Для того что бы создать базу данных необходимо иметь 4 класса:

1. класс Profile - это класс отвечающий за данные человека и будет иметь следующие свойства:

- Имя и Фамилия;

- Идентификационный номер;

- Уровень доступа – это список доступа, который персоналу разрешено находится;

- Статус персонала - свойство булевого типа, которое определяет статус персонала. Это сделано для того что бы при увольнении какого-либо сотрудника или истекания срока нахождении на территории оставались его данные в базе данных и просто отключали доступ;

- Фото – для распознавания лиц.

1. класс Lock - это класс отвечает за устройством исполнители:

- Идентификационный номер;

- ПОРТ соединения – порт, который соединен с устройством и служит его идентификационным номером;

- Местоположение – информация о местонахождении устройства;

- Фокальное расстояние (полученное экспериментально).

1. класс Activity - это класс отвечает за активность передвижение персонала по территории:

- Profile – это данные персонала, который прошел по устройству;

- дата и время;

- Lock – это данные устройства через который прошел персонал.

4) класс Permission – это класс отвечает для создания новые доступы:

- Название доступа;

- Дополнительная информация о доступе;

Таким образом составляется таблица в базу данных и получается следующая картина:

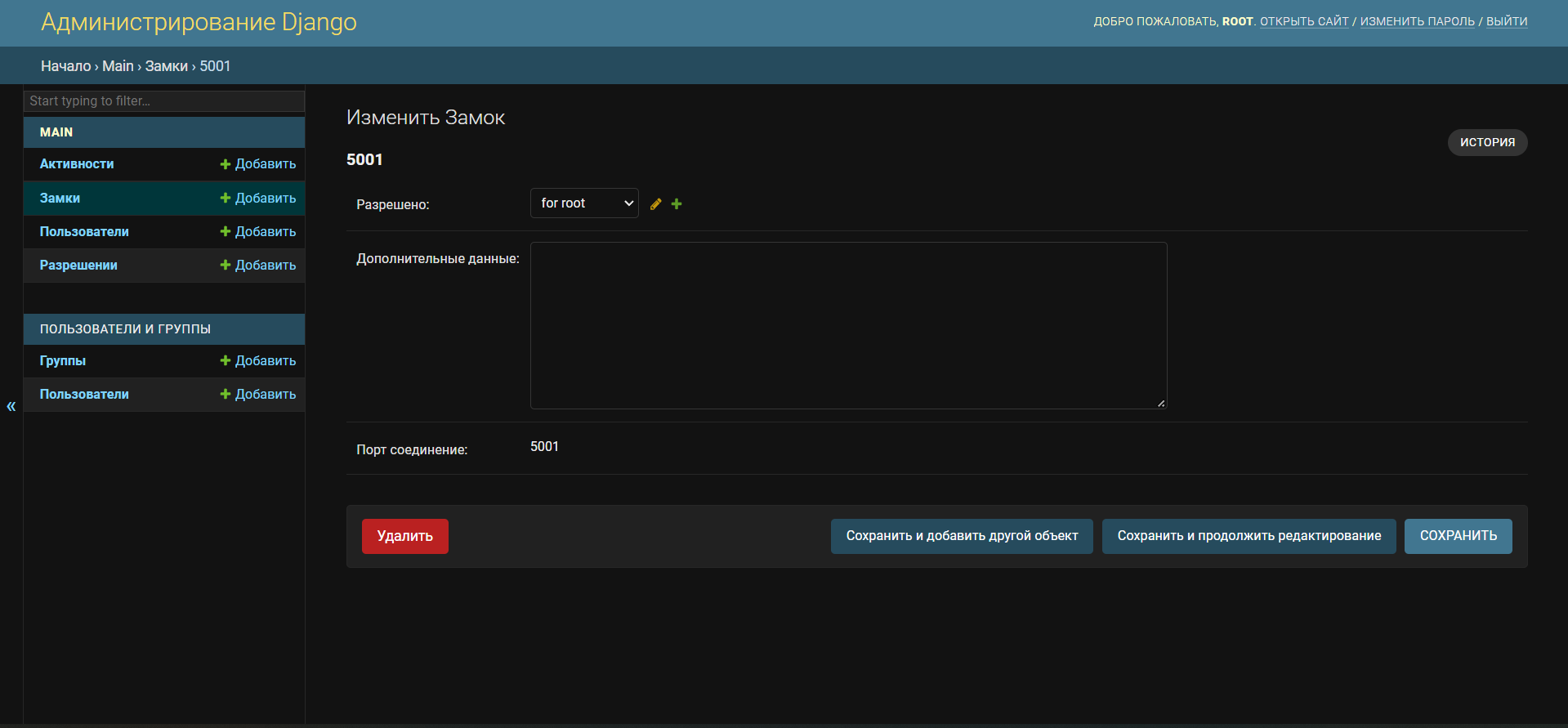


Рисунок 3.4 – Веб интерфейс. Главная страница.

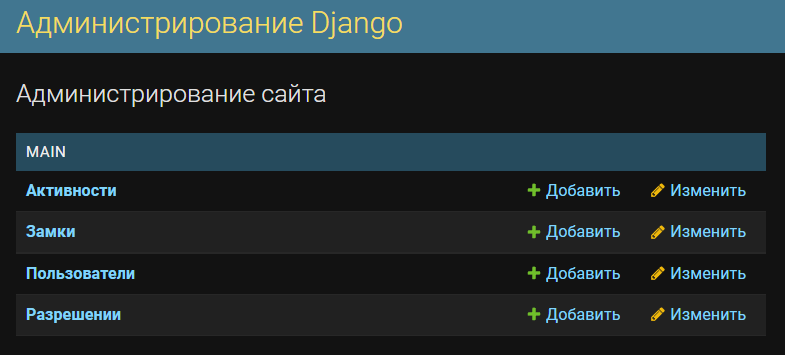


Рисунок 3.5 – Веб интерфейса панели администратора

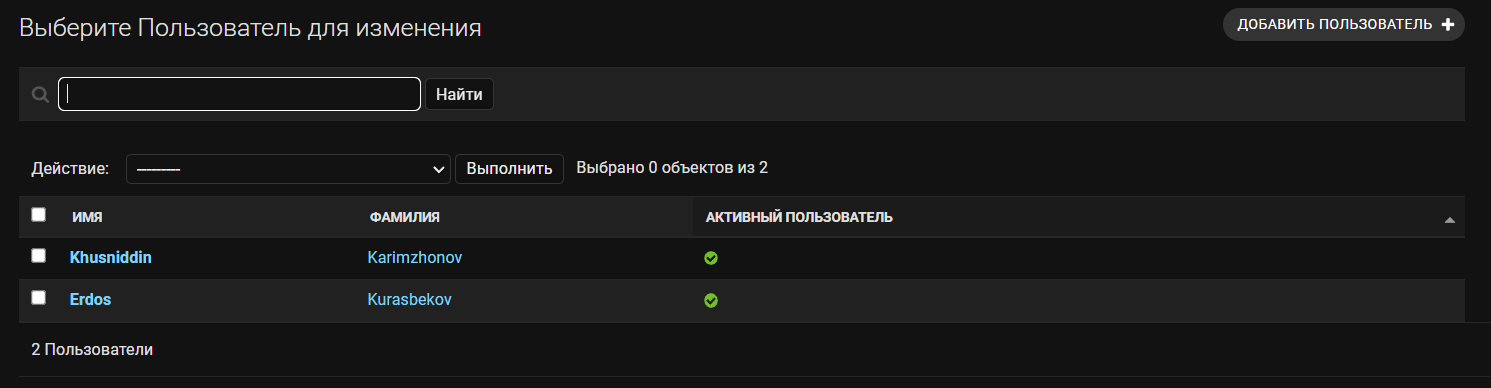


Рисунок 3.6 – Веб интерфейс списка персонала

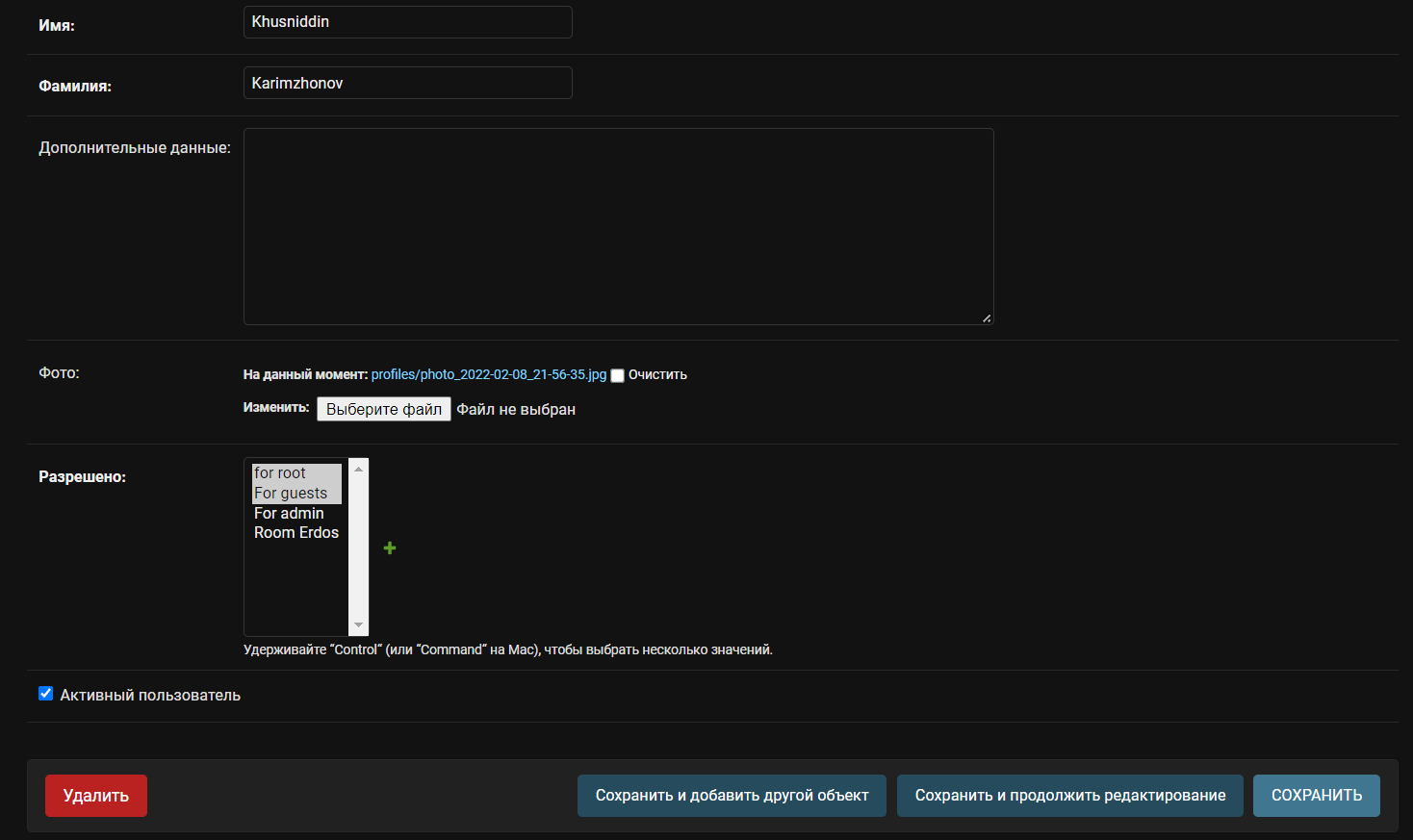


Рисунок 3.7 – Веб интерфейс управления данными персонала.

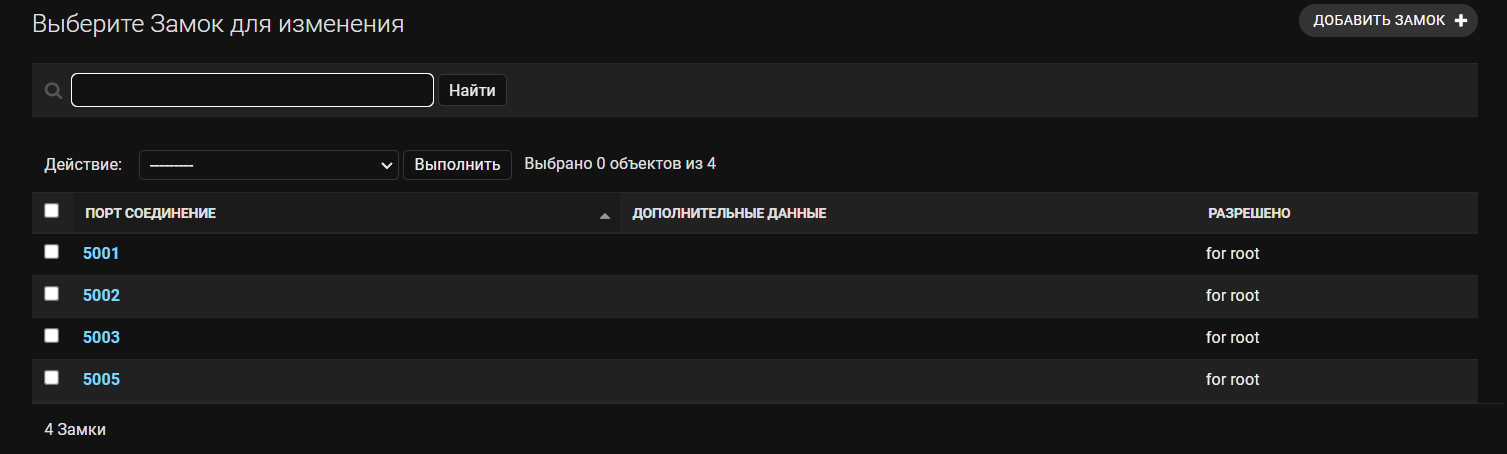


Рисунок 3.8 – Веб интерфейс списка устройств исполнителей

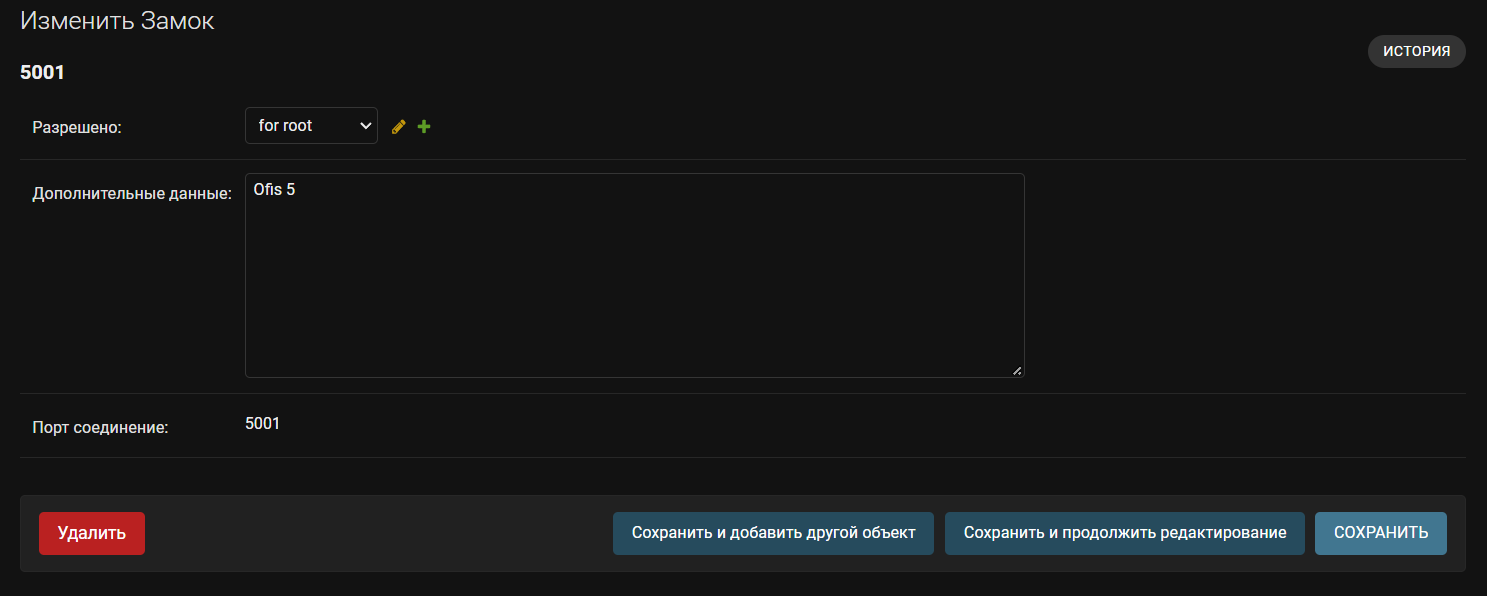
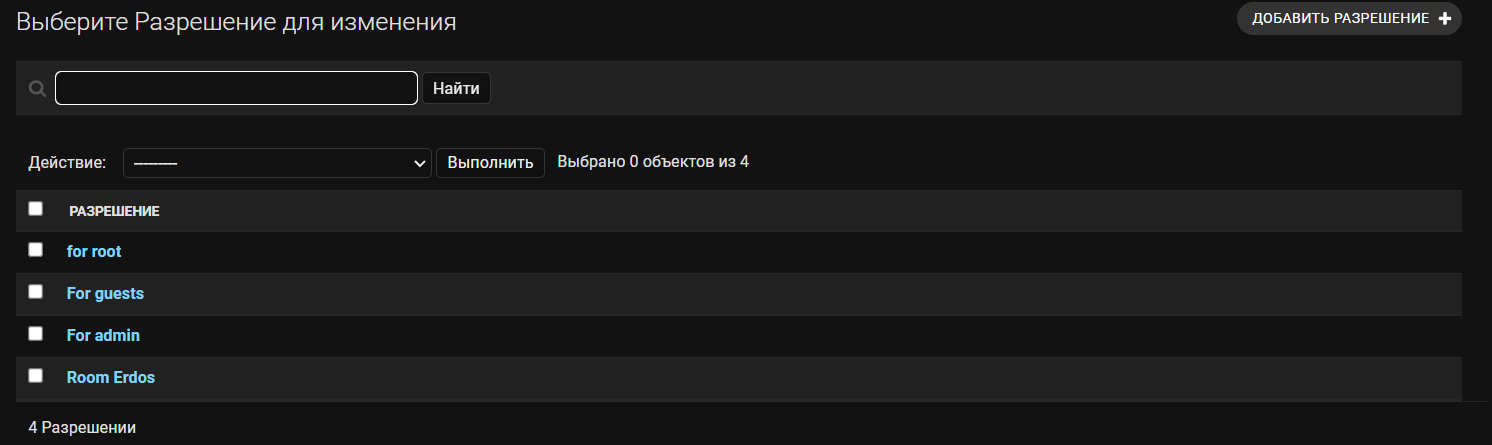
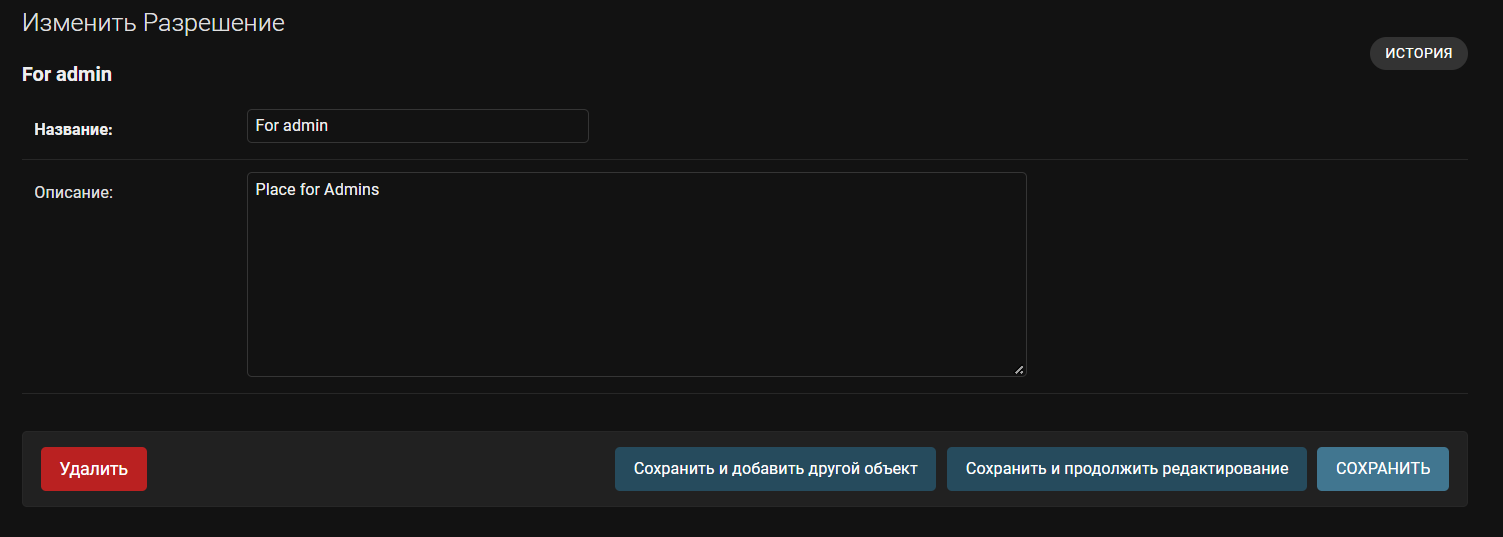
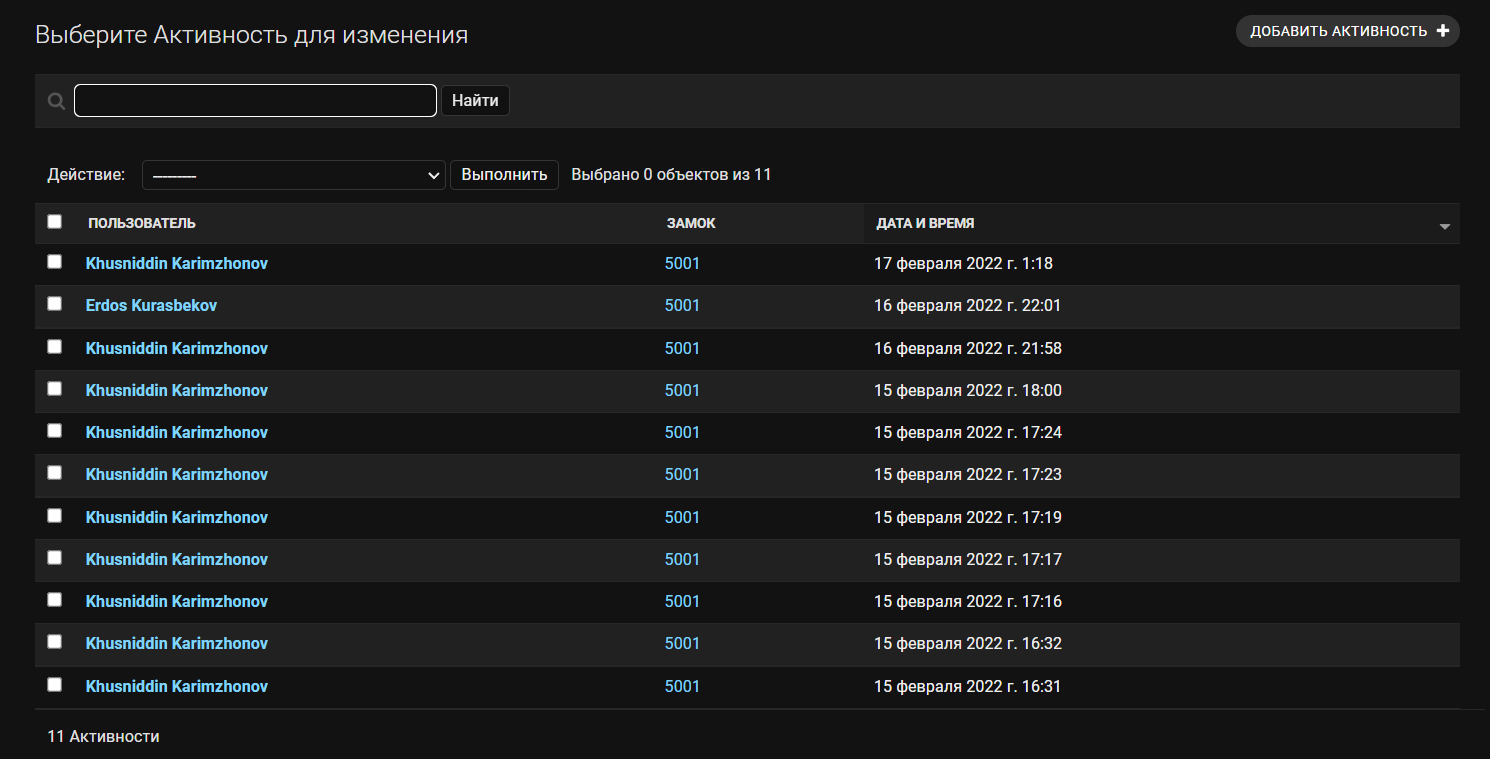
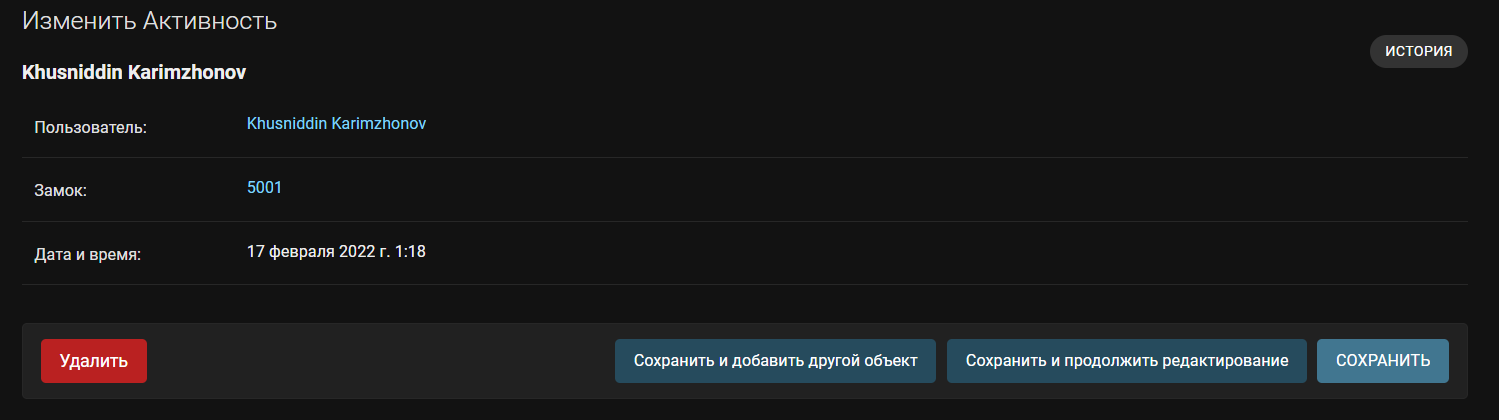


Рисунок 3.9 – Веб интерфейс корректировки данных устройств

 Рисунок 3.10 – Веб интерфейс списка доступных разрешений

 Рисунок 3.11 – Веб интерфейс корректировки разрешения

 Рисунок 3.12 – Веб интерфейс табло активности

 Рисунок 3.13 – Веб интерфейс дополнительной информации об активности

И интерфейс Django позволяет с легкости управлять этими данными и создавать временные пропуски для гостей.

* 1. **Графический интерфейс для отслеживания в реальном времени**

Автоматизированная система СКУД отлично работает в фоновом режиме, но при необходимости надо будет управляет с устройствами в ручном режиме. Для таких случаев разработан графический интерфейс, который предназначит для управления над устройствами исполнителями. Интерфейс представляет собой сетка кадров который получает сервер для обработки и кнопки который дает разрешение в ручном режиме (рис. 3.13 и рис. 3.14)

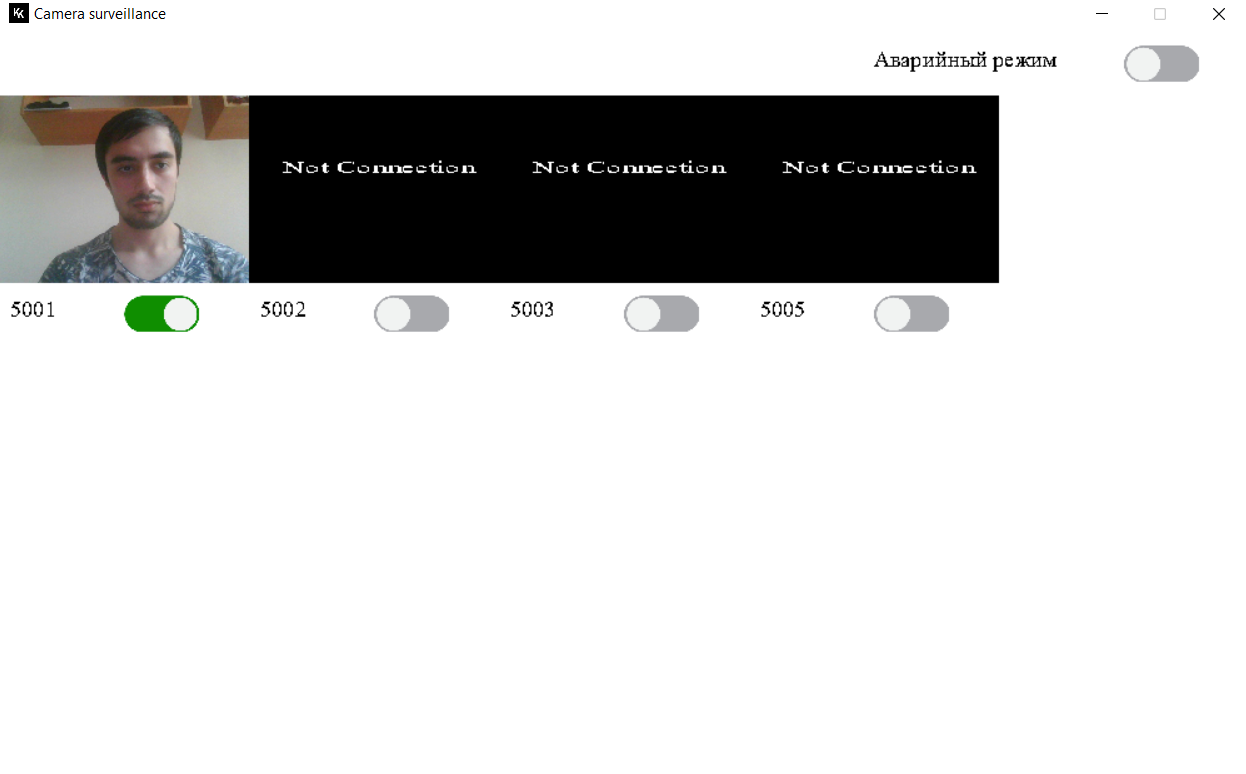


Рисунок 3.14 – Графический интерфейс

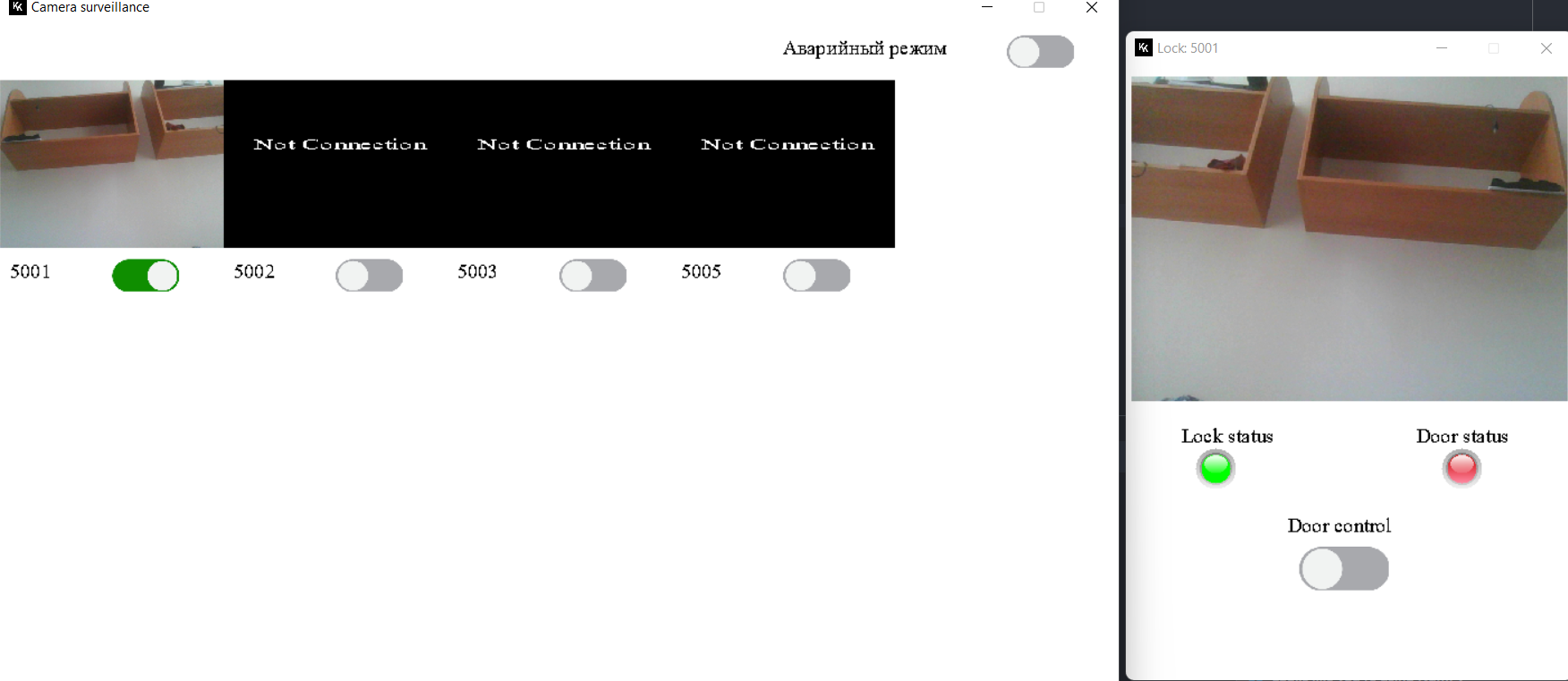


Рисунок 3.15 – Графический интерфейс Ручное управление

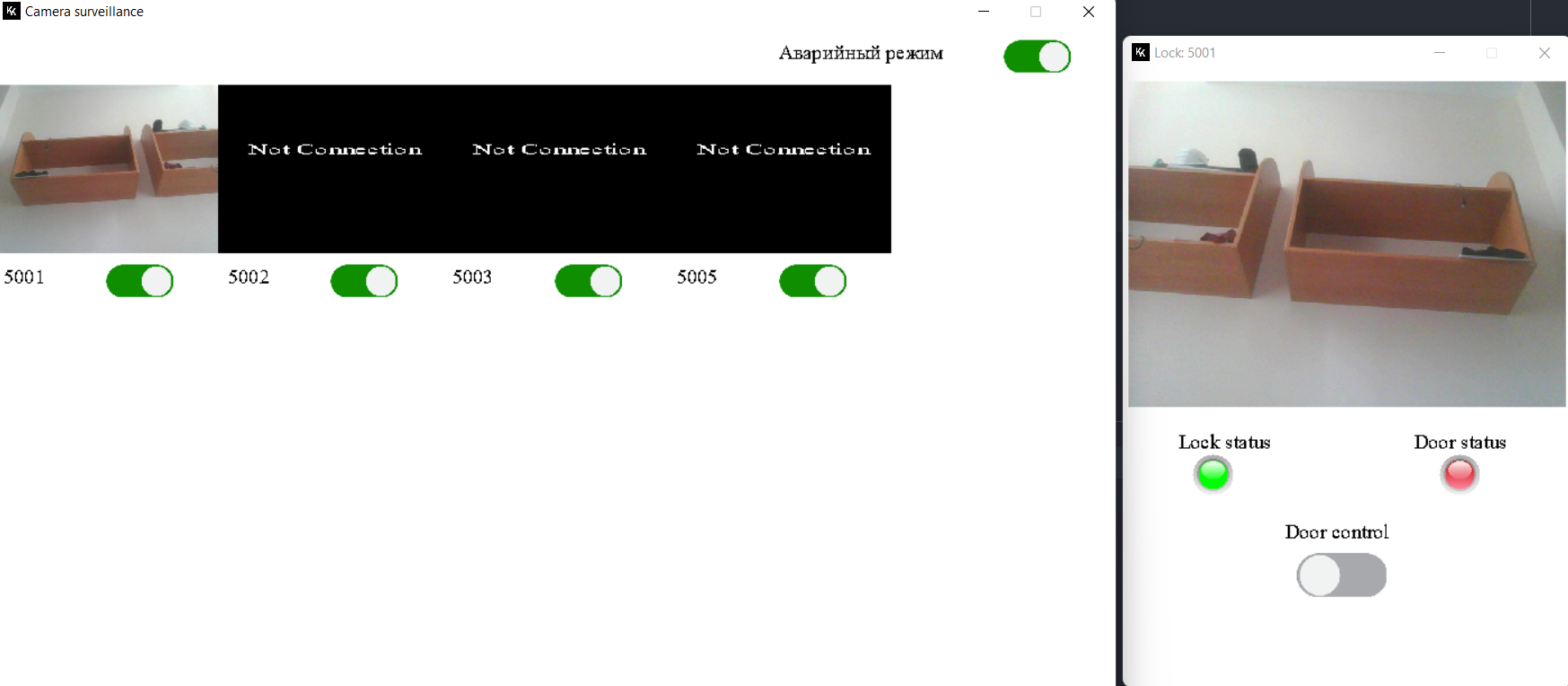


Рисунок 3.16 – Графический интерфейс. Аварийный режим.

В системе СКУД еще предусмотрено аварийный режим, при который будет открывается все замки, не смотря есть ли доступ у этого персонала или нет. (рис. 3.15)

**Выводы по разделу.** В данном разделе были реализованы два интерфейса:

* Веб интерфейс для управления базы данных.
* Графический интерфейс для управления устройств.

Также был разработан трех уровневый алгоритм аутентификации, который в практике показала себя достойно.

4 БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

**4.1 Общие положения**

В рамках данной работы была разработана пропускная система с помощью распознавания лиц и интерфейс для управления этой системы. В своей очереди этот интерфейс делится на два: веб интерфейс и графический интерфейс. Для этой цели был использован инструмент (IDE - Интегрированная среда разработки) PyCharm. Таким образом, в данном разделе ВКР бакалавра необходимо рассмотреть основные требования, которые накладываются на 3 интерфейсы программ.

В данной ВКР будет рассмотрены эргономические требования и принципы в соответствие с ГОСТ Р ИСО 9241, серия 100, которая описывает интерфейсы на основе программных средств.

**4.2** **Аспект эргономики программного обеспечения**

Стандарты по эргономике программного обеспечения содержат руководства по применению и детальное описание требований к проектированию и оценке пользовательского интерфейса интерактивной системы. Эти стандарты не направлены на стандартизацию пользовательского интерфейса, они лишь дают рекомендации по обеспечению пригодности использования пользовательского интерфейса и предотвращению появления конструкторских решений, которые с большой вероятностью вызовут проблемы с пригодностью использования.

На рисунке 4.1 показана роль стандартов по эргономике в анализе, проектировании и оценке программного обеспечения. Эти стандарты содержат рекомендации, которые применимы к широкому диапазону условий использования и которые необходимо приспособить к конкретным условиям использования проектируемой или перепроектируемой интерактивной системы. Не все рекомендации применяют в любых условиях использования. Для конкретных проектных и конструкторских решений требования и рекомендации стандартов по эргономике программного обеспечения помогают обеспечить подходящие уровни пригодности использования и доступности.

Требования и рекомендации, содержащиеся в этих стандартах, должны быть оценены с точки зрения их применимости в определенных условиях использования и применены в соответствии с результатами этой оценки.

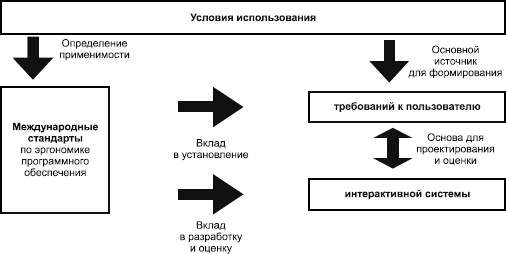


Рис. 4.1 – Роль стандартов по эргономике программного обеспечения в анализе, проектировании и оценке

Настоящий стандарт позволяет пользователям стандартов по эргономике программного обеспечения определить стандарты, являющиеся наиболее важными для разработки программного обеспечения, получить обзор содержания этих стандартов, понять их роль в установлении требований к пользователю, проектированию и оценке пользовательских интерфейсов и понять взаимосвязь между различными стандартами.

Стандарты по эргономике программного обеспечения применимы ко всем программным компонентам интерактивной системы, которые оказывают влияние на пригодность использования, включая:

- прикладные программы (в том числе интернет-приложения);

- операционные системы;

- встроенное программное обеспечение;

- программные средства проектирования;

- вспомогательные технологии.

Диапазон стандартов, рассмотренных в настоящем стандарте, включает основные международные стандарты по эргономике программного обеспечения и международные стандарты по процессам и методам, относящимся к эргономике программного обеспечения.

Настоящий стандарт предоставляет сводную информацию по стандартам, которые могут быть использованы представителями следующих профессий:

- эргономистами;

- специалистами в области пригодности использования;

- проектировщиками;

- руководителями проектов;

- менеджерами;

- работниками или их представителями;

- потребителями или их представителями;

- поставщиками;

- работниками органов по сертификации.

Согласно стандарту ГОСТ 9241, серия 100, выделим главные требования к программному обеспечению интерфейса:

1. интерфейс должен быть пригоден для выполнения задач, управляемым, информативным и удобным для обучения (Стандарт ИСО 9241-110);
2. представляемая информация должна быть четкой, распознаваемой, лаконичной, разборчивой и понятной (Стандарт ИСО 9241-12);
3. наличие элементов пользовательского интерфейса, помогающих пользователю в освоении и использовании системы (Стандарт ИСО 9241-130);
4. для поддержи диалога между пользователем и программным обеспечением должны существовать технические приемы, такие как: меню, команды, непосредственное управление, диалоги вопроса и ответа (Стандарт ИСО 9241-140);
5. наличие диалогов на основе меню и командных диалогов (Стандарт ИСО 9241-14,15);
6. доступность программного обеспечения для широкого диапазона пользователей (Стандарт ИСО 9241-171) [[9]](#_СПИСОК_ИСПОЛЬЗОВАННЫХ_ИСТОЧНИКОВ).

## **4.3 Соответствие разработанного ПО требованиям ГОСТ Р ИСО 9241**

В ходе выполнения данной работы была разработана два интерфейса:

* Веб интерфейс (для работы база данных) представляет собой удобный, практичный и интуитивно понятным видам.

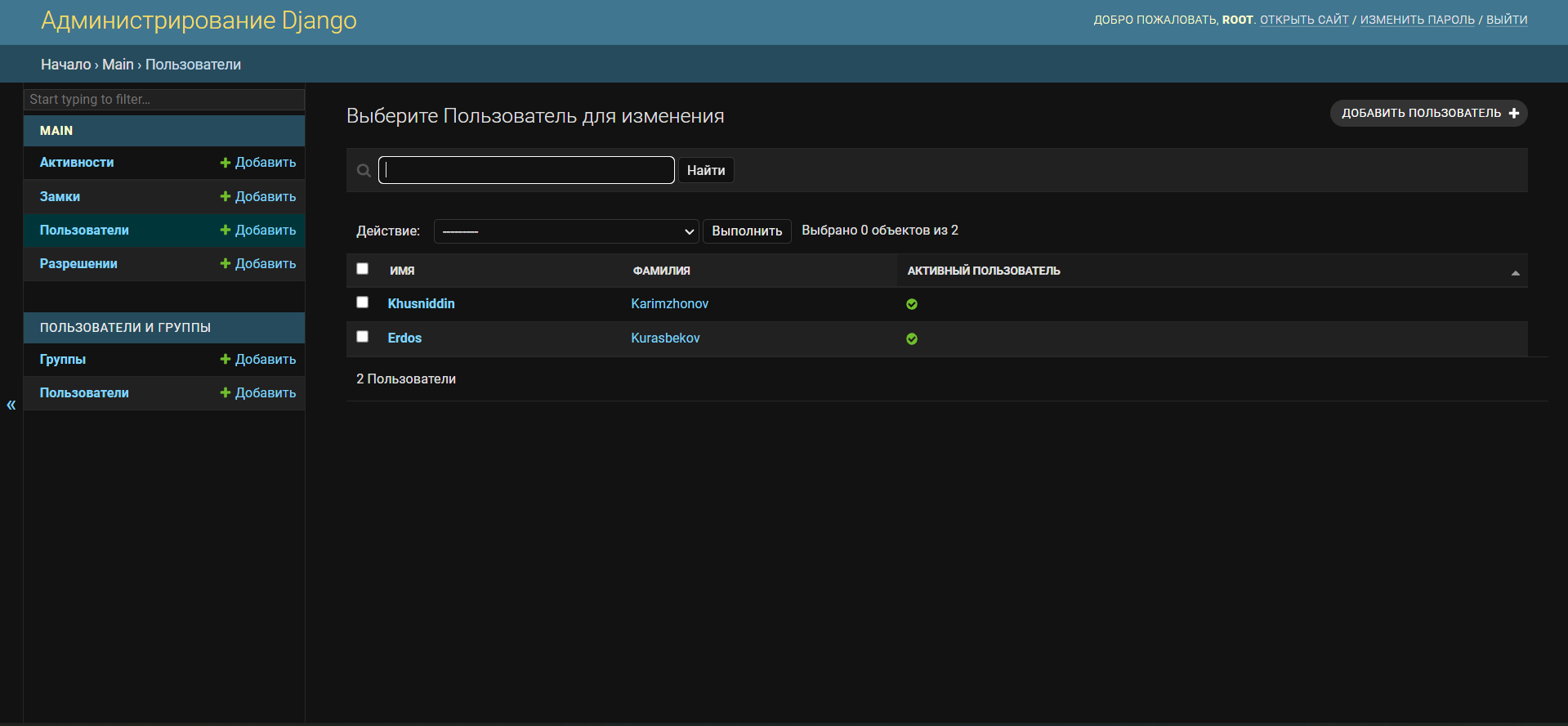


Рис. 4.2 – Веб интерфейс. Список персоналов.

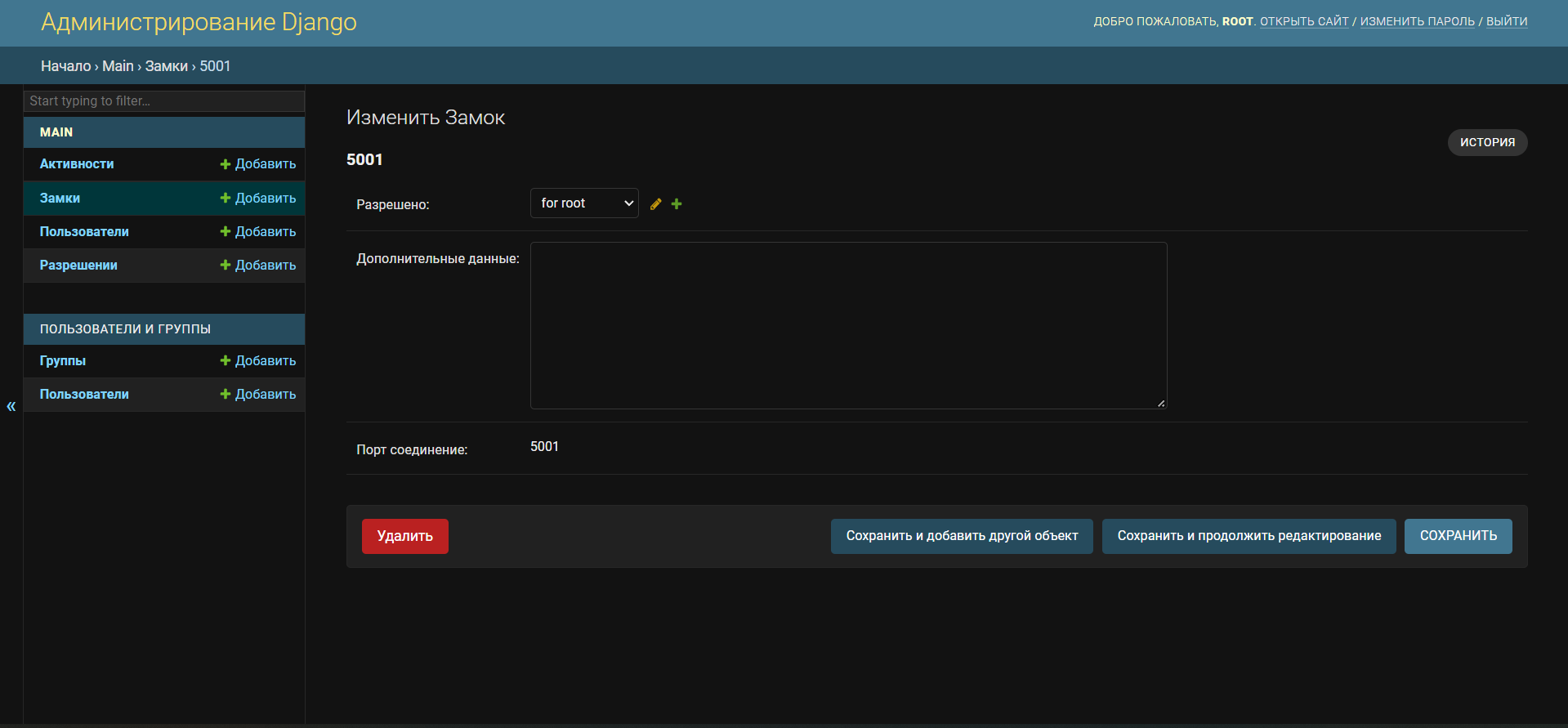


Рис. 4.3 – Веб интерфейс. Редактирование данных

Интерфейс разделен на несколько блоков:

1. Верхний панель – верхном панели представлен имя пользователя, который при нажатии всплывает окно с полной информацией. Также в этом панели находится навигация, с помощью которого можно узнать в какой странице находится пользователь.
2. Боковая панель – в этом панели находится все основные раздели данного веб интерфейса. Для визуального комфорта, при наведении подсвечивается все кнопки.
3. Основная рабочая область – в этой области выполняется все основные действия пользователя (редактирование данных пользователя, изменения параметра устройств и наблюдение за активности). В этой области все кнопки подсвечиваются смотря какую функцию она выполняет (красный – удаления данных, синий – сохранение, обновление и создание данных)

* Графический интерфейс (для управление устройствами) представляет собой одна целая рабочая поверхность, в которой находится изображение получаемых кадров, кнопка для управления устройствами. Также находится кнопка для аварийного режима, при включении, которого открывается все устройства. Это сделано для того, чтобы при экстренных ситуациях персоналом можно было покинуть помещение без препятствия. И все кнопки имеет два характерного света (зеленый – означающий активность кнопки и серий – деактивность)

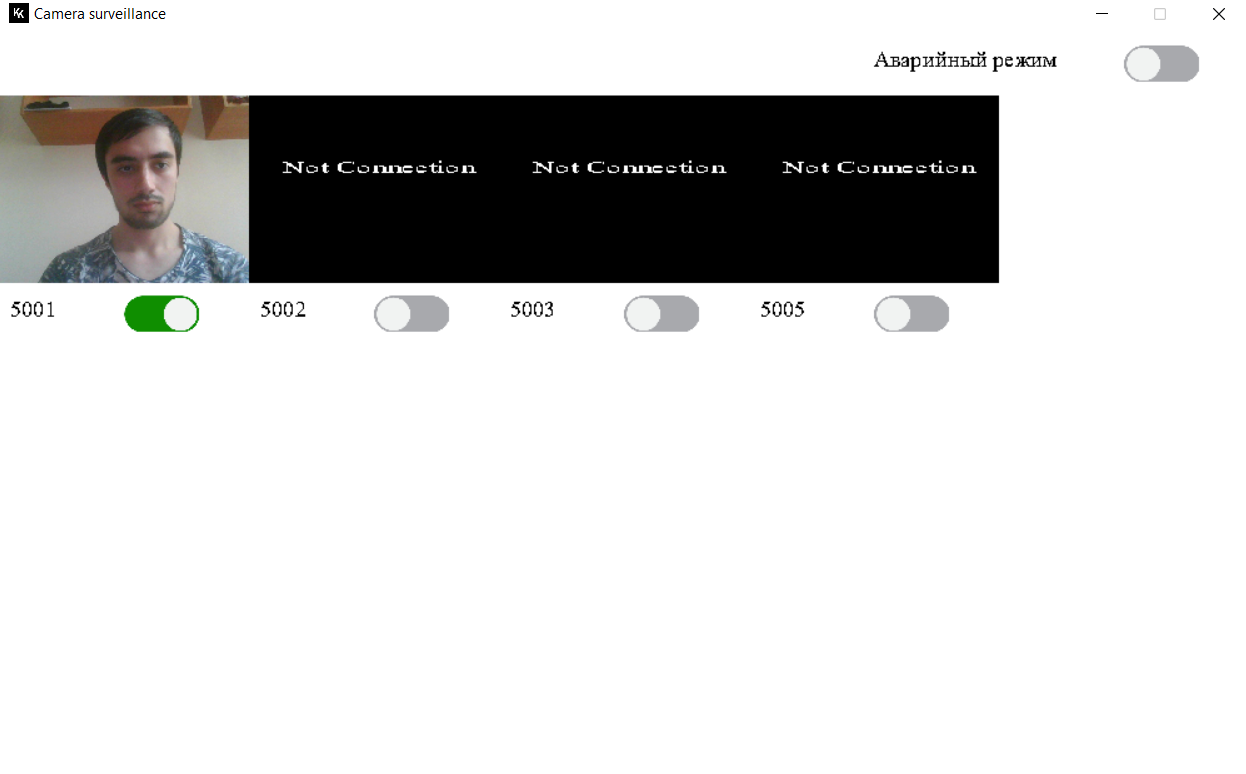


Рис. 4.4 – Графический интерфейс

Исходя из этого можно сказать что, оба интерфейса отвечает всем требованием ГОСТ Р ИСО 9241 серия 100.

## **4.4 Соответствие использованного ПО требованиям ГОСТ Р ИСО 9241**

PyCharm — это среда разработки для языка программирования Python. Разработка IDE ведется в компании JetBrains флагманским продуктом, которой является IntelliJ IDEA, на его основе разработаны остальные продукты JetBrains в том числе и PyCharm.

В среде PyCharm пользователю предоставлена свобода выбора в самостоятельной настройке вида интерфейса и его окон, свобода расположения окон. Для удобства использования интерфейса PyCharm разделен на несколько отличающихся друг от друга по функционалу областей.

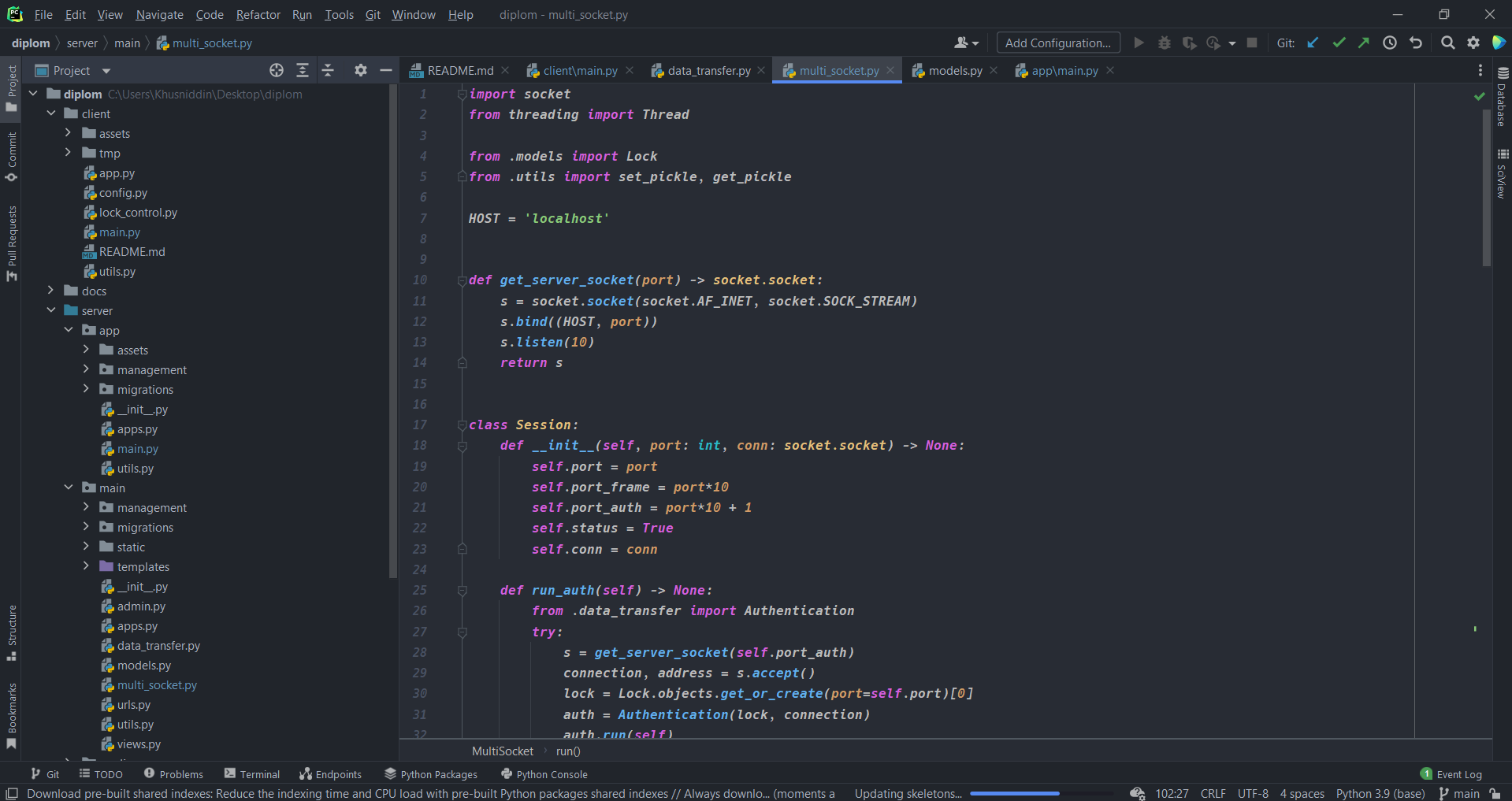


Рисунок 4.5 – Интерфейс PyCharm

Интерфейс содержит следующие элементы:

* Блок код – основная рабочая область, в которой идет написание кода. Название функций и прочее подсвечено разными цветами. Здесь же указан путь до файла, а чуть выше расположены вкладки-файлы, по которым можно перемещаться. С помощью них мы можем добавлять не просто один файл с кодом, а целый проект.
* Активное меню – В котором находятся главные функции редактора. Вкладка проводника открывается по умолчанию при запуске программы. В неё выводятся список открытых файлов и каталог открытой папки.
* Строка состояния – В этой панели находятся команды “Git”, “TODO”, “Проблемы”, “Консоль откладки” и “Терминал”. С помощью этих команд можно узнать количество ошибок, которые возникли в ходе выполнения кода, отфильтровать журнал и очистить его.

При этом пользователь имеет возможность сам настраивать вид интерфейса. Также важно размещение элементов. Большую часть интерфейса занимает блок кода, так как вся главная работа осуществляется именно в этом блоке. Слева расположены активное меню, в нем находится информация, которая менее важна для пользователя, но к которой можно обратиться в любой момент времени.

Так же в PyCharm представлена удобная в пользовании и простая в понимании панель управления, некоторые кнопки представляют собой изображение, указывающее назначение этой кнопки. Также для помощи в изучении данной системы существует кнопка Справка, в которой есть видеоруководство, документация, поиск и запросов функции, советы и рекомендации. Это дает удобство и легкость управления пользователю.

Таким образом, работа в среде PyCharm отвечает всем требованием, учитывает человеческие особенности восприятия и работы, что позволяет достигнуть достаточно хороших результатов при работе в данной среде.

**Выводы по разделу.** В данном разделе изучены области применения аспектов эргономики, рассмотрена эргономика использованного и разработанного программного обеспечения. После сравнительного анализа сделан вывод, что программное обеспечение для исследования системы удовлетворяем стандартам эргономичного использования (ГОСТ 9241)

# ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломного проекта была разработана система  
контроля и управления доступом персонала, связанных с распознаванием лиц. Были рассмотрены разные алгоритмы распознавания лиц, архитектура передачи данных, необходимых для их реализации. Также были разработаны два интерфейса для управления такой системы.

В роле устройство был разработан симулятор, который представляет собой графический интерфейс показывающий состояние устройств.

Система себя оправдает для любых организаций и предприятий, с любым количеством персоналов, где важно значительно снизить человеческий фактор, повысить актуальность информации о перемещение персонала и всегда иметь возможность выполнить быстрый анализ и получить отчет.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Дипломная работа о «Система контроля и управления доступом автотранспорта» [Электронный ресурс] URL: <https://elib.pnzgu.ru/files/eb/doc/kJqTvqTWUAW.pdf> (дата обращения: 30.04.2022)
2. Установка систем биометрического распознаванию лиц [Электронный ресурс] URL: <https://ipsdi.ru/biometricheskogo-raspoznavaniju-lic> (дата обращения: 27.01.2022)
3. Анализ существующих подходов к распознаванию лиц [Электронный ресурс] [URL: https://habr.com/ru/company/synesis/blog/238129/](URL:%20https://habr.com/ru/company/synesis/blog/238129/) (дата обращения: 23.05.2022)
4. Как работает распознавание лиц? Разбор [Электронный ресурс] URL: <https://droider.ru/post/kak-rabotaet-raspoznavanie-licz-razbor-07-07-2021/> (дата обращения: 27.01.2022)
5. Официальная сайт OpenCV [Электронный ресурс] URL: <https://opencv.org/> (дата обращения: 28.01.2022)
6. Официальная сайт Face Recognition [Электронный ресурс] URL: <https://github.com/ageitgey/face_recognition> (дата обращения: 28.01.2022)
7. Официальная сайт Django [Электронный ресурс] URL: <https://www.djangoproject.com/> (дата обращения: 28.01.2022)
8. Принцип сокетов [Электронный ресурс] URL: <https://lecturesnet.readthedocs.io/net/low-level/ipc/socket/intro.html> (дата обращения: 28.01.2022)
9. Национальный стандарт российской федерации [Электронный ресурс] URL: <https://docs.cntd.ru/document/1200097753> (дата обращения: 23.05.2022)

# ПРИЛОЖЕНИЕ А

**Код получения потоковой информации**

import socket  
from threading import Thread  
  
from .models import Lock  
from .utils import set\_pickle, get\_pickle  
  
HOST = 'localhost'  
  
  
def get\_server\_socket(port) -> socket.socket:  
 s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
 s.bind((HOST, port))  
 s.listen(10)  
 return s  
   
  
class Session:  
 def \_\_init\_\_(self, port: int, conn: socket.socket) -> None:  
 self.port = port  
 self.port\_frame = port\*10  
 self.port\_auth = port\*10 + 1  
 self.status = True  
 self.conn = conn  
  
 def run\_auth(self) -> None:  
 from .data\_transfer import Authentication  
 try:  
 s = get\_server\_socket(self.port\_auth)  
 connection, address = s.accept()  
 lock = Lock.objects.get\_or\_create(port=self.port)[0]  
 auth = Authentication(lock, connection)  
 auth.run(self)  
 print(f'Port closed: {self.port\_auth}')  
 connection.close()  
 s.close()  
 except IndexError:  
 pass  
  
 def run\_frames(self) -> None:  
 from .data\_transfer import Frame  
 try:  
 s = get\_server\_socket(self.port\_frame)  
 connection, address = s.accept()  
 lock = Lock.objects.get\_or\_create(port=self.port)[0]  
 frame = Frame(lock, connection)  
 frame.run(self)  
 print(f'Port closed: {self.port\_frame}')  
 connection.close()  
 s.close()  
 except IndexError:  
 pass  
  
 def run(self) -> None:  
 Thread(target=self.run\_frames).start()  
 Thread(target=self.run\_auth).start()  
 self.conn.close()  
  
  
class MultiSocket:  
 from server.settings import BASE\_DIR  
  
 PORT = 5000  
 filename = f'{BASE\_DIR}/tmp/start.pickle'  
 active\_ports = f'{BASE\_DIR}/tmp/active.pickle'  
  
 @classmethod  
 def \_set\_active\_ports(cls, value):  
 set\_pickle(cls.active\_ports, value)  
  
 @classmethod  
 def get\_active\_ports(cls):  
 try:  
 return get\_pickle(cls.active\_ports)  
 except FileNotFoundError:  
 cls.\_set\_active\_ports([])  
 return []  
  
 @classmethod  
 def add\_active\_ports(cls, port: int):  
 ports = cls.get\_active\_ports()  
 ports.append(port)  
 cls.\_set\_active\_ports(ports)  
  
 @classmethod  
 def remove\_active\_ports(cls, port: int):  
 try:  
 ports = cls.get\_active\_ports()  
 index = ports.index(port)  
 ports.pop(index)  
 cls.\_set\_active\_ports(ports)  
 except ValueError:  
 pass  
  
 @classmethod  
 def get\_status(cls):  
 return get\_pickle(cls.filename)  
  
 @classmethod  
 def set\_status(cls, status: bool):  
 set\_pickle(cls.filename, status)  
  
 def run(self) -> None:  
 s = get\_server\_socket(self.PORT)  
 print('Server started')  
 while True:  
 connection, address = s.accept()  
 if not self.get\_status():  
 connection.close()  
 s.close()  
 self.\_set\_active\_ports([])  
 break  
 print('Client connected: ', address)  
 host, port = address  
 # Active Ports Set  
 self.add\_active\_ports(port)  
  
 ses = Session(port, connection)  
 Thread(target=ses.run).start()

# ПРИЛОЖЕНИЕ Б

**Код аутентификации**

from time import time  
import cv2  
import pickle  
import numpy as np  
  
from threading import Thread  
from socket import socket  
from server import settings  
from .models import Lock, Activity, Profile  
from .multi\_socket import MultiSocket  
from .utils import get\_pickle, set\_pickle  
from app.utils import get\_emergency\_control\_status

class Authentication(Client):  
 def \_out\_while\_errors(self):  
 self.save\_auth((0, 0))  
  
 def save\_auth(self, status):  
 set\_pickle(self.lock.get\_last\_auth\_path(), status)  
  
 def get\_auth(self):  
 try:  
 return get\_pickle(self.lock.get\_last\_auth\_path())  
 except FileNotFoundError:  
 self.save\_auth((0, 0))  
 return 0, 0  
  
 @staticmethod  
 def check\_distance(frame: np.array) -> int:  
 from mediapipe.python.solutions.face\_mesh import FaceMesh  
  
 def get\_prefigure(lx, ly, rx, ry):  
 return ((lx - rx) \*\* 2 + (ly - ry) \*\* 2) \*\* (1 / 2)  
  
 def get\_distance(dis\_eyes):  
 return settings.FOCAL\_DEFAULT \* settings.EYES\_DISTACE\_DEFAULT / dis\_eyes  
  
 def get\_focal(dis\_eyes: float, d: float):  
 return d \* dis\_eyes / settings.EYES\_DISTACE\_DEFAULT  
  
 status = 0  
 try:  
 h, w, \_ = frame.shape  
 coordinates = []  
  
 detector = FaceMesh(max\_num\_faces=10)  
 frame\_rgb = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)  
 landmarks = detector.process(frame\_rgb).multi\_face\_landmarks  
 if landmarks:  
 for point in landmarks:  
 left\_eye = point.landmark[23]  
 right\_eye = point.landmark[253]  
 lx, ly = int(left\_eye.x \* w), int(left\_eye.y \* h)  
 rx, ry = int(right\_eye.x \* w), int(right\_eye.y \* h)  
 coordinates.append(((lx, ly), (rx, ry)))  
 if coordinates:  
 left, right = coordinates[0]  
 min\_dis\_eyes = get\_prefigure(\*left, \*right)  
 for (lx, ly), (rx, ry) in coordinates:  
 dis = get\_prefigure(lx, ly, rx, ry)  
 if min\_dis\_eyes > dis:  
 min\_dis\_eyes = dis  
  
 distance\_to\_camera = get\_distance(min\_dis\_eyes)  
 if distance\_to\_camera < settings.MAX\_AUTH\_DISTANCE:  
 status = 1  
 except AttributeError:  
 pass  
 return status  
  
 @staticmethod  
 def compare\_face(frame: np.array) -> tuple:  
 import face\_recognition as fr  
  
 status = 0  
 profile = []  
 true\_face\_encs, profiles = Profile.get\_profiles\_encs()  
  
 frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)  
 frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)  
  
 face\_locs = fr.face\_locations(frame)  
 face\_encs = fr.face\_encodings(frame, face\_locs)  
  
 for enc in face\_encs:  
 flags = fr.compare\_faces(true\_face\_encs, enc)  
  
 for i, flag in enumerate(flags):  
 if flag:  
 status = 1  
 profile.append(profiles[i])  
  
 return status, profile  
  
 @staticmethod  
 def check\_permission(profiles: list[Profile], lock) -> int:  
 status = 0  
 for profile in profiles:  
 if profile.check\_permissions(lock):  
 status = 1  
 return status  
  
 def auth(self, frame: np.array):  
 profiles = None  
 status = self.check\_distance(frame)  
  
 if status:  
 status, profiles = self.compare\_face(frame)  
  
 if status:  
 status = self.check\_permission(profiles, self.lock)  
  
 if status:  
 Thread(target=Activity.add\_activity, args=(profiles, self.lock)).start()  
 \_, app\_status = self.get\_auth()  
 self.save\_auth((status, app\_status))  
  
 def main(self) -> None:  
 status, app\_control = self.get\_auth()  
 emergency\_control\_status = get\_emergency\_control\_status()  
  
 if emergency\_control\_status:  
 self.conn.recv(4096)  
 answer = pickle.dumps(emergency\_control\_status)  
 self.conn.send(answer)  
 elif status or app\_control:  
 t0 = time()  
 while time() - t0 < settings.AUTH\_DELAY:  
 self.conn.recv(4096)  
 answer = pickle.dumps(1)  
 self.conn.send(answer)  
 self.save\_auth((0, 0))  
 else:  
 self.conn.recv(4096)  
 answer = pickle.dumps(status)  
 self.conn.send(answer)  
  
 frame = cv2.imread(self.lock.get\_last\_frame\_path(), cv2.IMREAD\_ANYCOLOR)  
 self.auth(frame)