**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра CАУ**

отчет

**по практике**

Тема: Управление пропускной системой с помощью распознавания лиц

|  |  |  |
| --- | --- | --- |
| Студент гр. 8491 |  | Каримжонов Х.Т. |
| Руководитель |  | Никоза А.В. |

Санкт-Петербург

2022

**ЗАДАНИЕ**

**на практику**

|  |  |  |
| --- | --- | --- |
| Студент Каримжонов Х.Т. | | |
| Группа 8491 | | |
| Тема практики: Управление пропускной системой с помощью распознавания лиц | | |
| Задание на практику:  Составить план для ВКР и формулировать решаемых задач. | | |
| Сроки прохождения практики: 13.01.2022 – 13.02.2022 | | |
| Дата сдачи отчета: | | |
| Дата защиты отчета: | | |
|  | | |
| Студент |  | Каримжонов Х.Т. |
| Руководитель |  | Никоза А.В |

**содержание**

|  |  |  |
| --- | --- | --- |
|  | Введение | 4 |
| 1. | Общее положение | 5 |
| 2. | Реализация алгоритма распознавания лиц в среде Python. | 6 |
| 2.1. | Принцип работы распознавания лиц | 6 |
| 2.2. | Библиотека OpenCV | 10 |
| 2.3. | Библиотека Face Recognition | 11 |
| 2.4. | Нахождение расстояния между камерой и лицом | 11 |
| 3. | Разработка серверной части | 13 |
| 3.1. | Разработка базы данных в Django | 13 |
| 4. | Связь между сервером и клиентом | 15 |
| 4.1. | Сокетное программирование | 15 |
| 5. | Управления замком | 17 |
|  | Заключение | 18 |
|  | Список использованных источников | 19 |

**введение**

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.

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.

В ходе выполнения данной работы ознакомимся, алгоритм распознавания лиц, разработка серверной части для сохранения данных и принцип управления электрическим замком.

1. **Общее положение**

Процесс аутентификации имеет 2 подсистемы, которые представлено в рисунке.

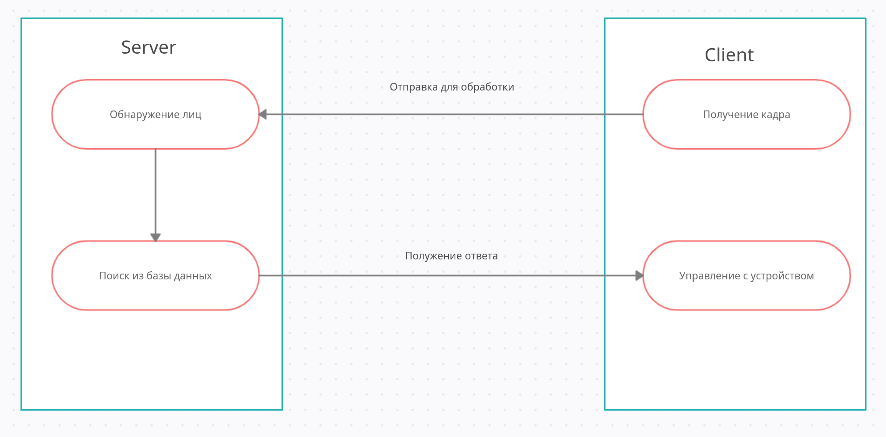


Рис.1 Схема аутентификации

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. Эта микросхема программируется на языке С++ или Python.

1. **Реализация алгоритма распознавания лиц в среде Python.**
   1. **Принцип работы распознавания лиц**

#### 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 (Рис.2), они называются примитивами Хаара:

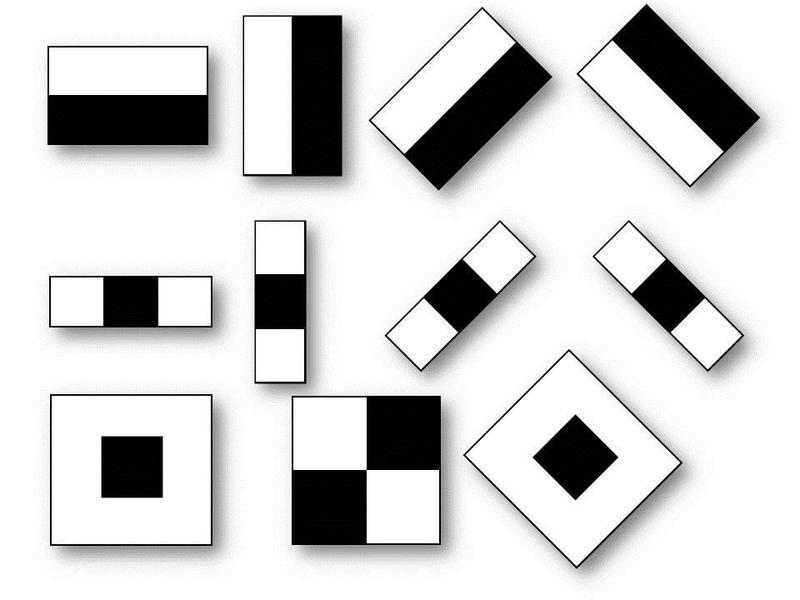


Рис.2 Примитивы Хаара

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.

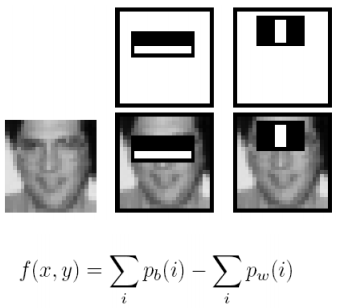


рис.3 Пример использование примитивы Хаара

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.

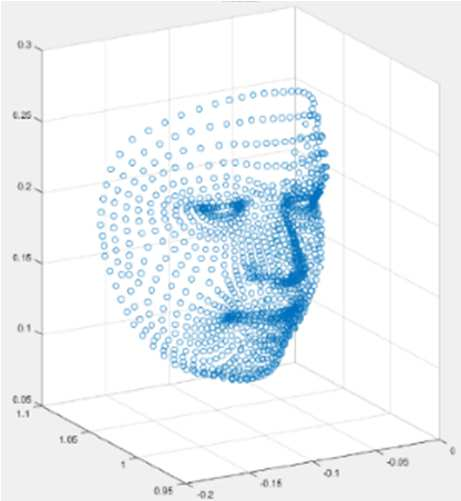


рис.4 Сетка лица

#### 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.

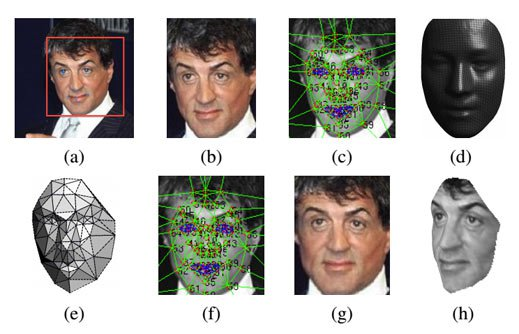


рис5. пример исправление искажение

#### 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.

#### Идентификация. Ну а дальше остаётся сравнить полученный вектор с базой других векторов. И готово. Система вас идентифицировала.

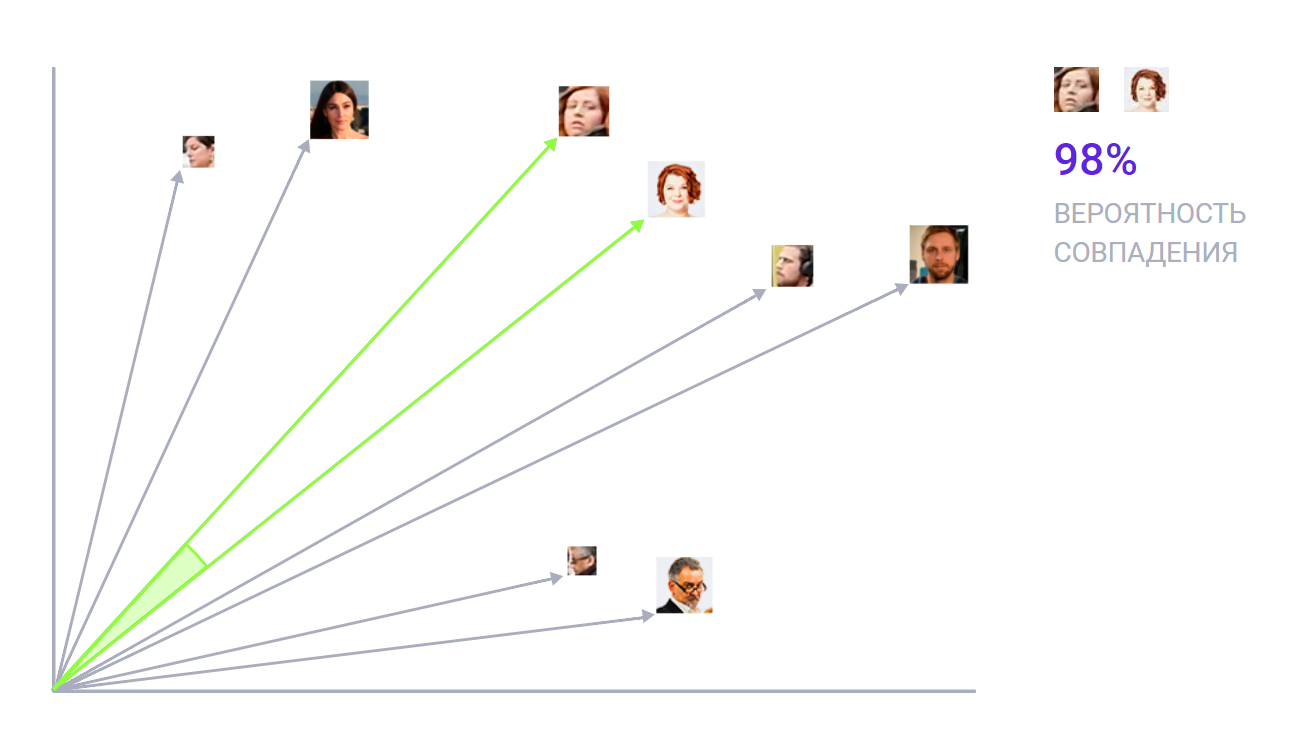


рис6. Сравнение с базой данных

В рамках данной работы будем использовать язык програмирования Python и его 2 многофункциональные библиотеки OpenCV и Face Recognition, который для нас будет выполнять необходимые для нас процессы

**2.2. Библиотека OpenCV**

OpenCV (Библиотека компьютерного зрения с открытым исходным кодом) — это библиотека программного обеспечения для компьютерного зрения и машинного обучения с открытым исходным кодом. OpenCV был создан, чтобы обеспечить общую инфраструктуру для приложений компьютерного зрения и ускорить использование машинного восприятия в коммерческих продуктах. Будучи продуктом с лицензией BSD, OpenCV позволяет предприятиям легко использовать и изменять код.

Библиотека содержит более 2500 оптимизированных алгоритмов, включая полный набор как классических, так и современных алгоритмов компьютерного зрения и машинного обучения. Эти алгоритмы можно использовать для обнаружения и распознавания лиц, идентификации объектов, классификации действий человека в видео, отслеживания движений камеры, отслеживания движущихся объектов, объединения изображений для получения высокого разрешения. изображения всей сцены, находить похожие изображения из базы данных изображений. Библиотека широко используется в компаниях, исследовательских группах и государственных органах.

Он имеет интерфейсы C++, Python, Java и MATLAB и поддерживает Windows, Linux, Android и Mac OS. OpenCV в основном ориентирован на приложения машинного зрения в реальном времени и использует инструкции MMX и SSE, когда они доступны. Полнофункциональные интерфейсы CUDA и OpenCL сейчас активно разрабатываются.

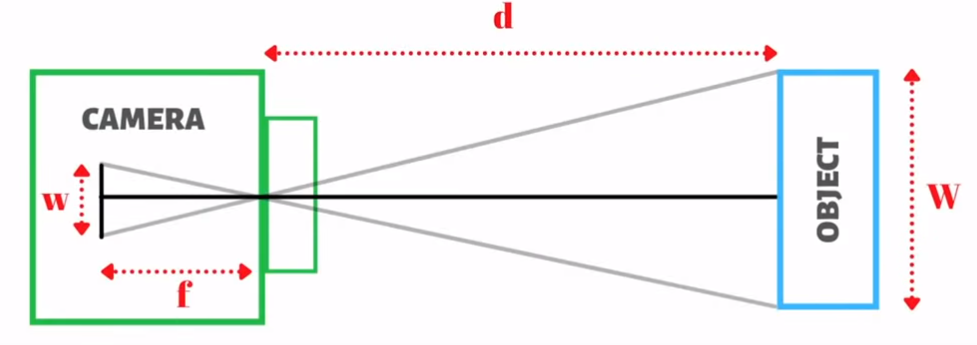
А мы будем использовать OpenCV для того что бы получить кадри с камеры в реальном времени и отправить эту изображению на обработку, где нас будет ждать библиотека Face recognition

**2.3. Библиотека Face Recognition**

Face Recognition - это Open Source продукт разработано программистом Адам Геитгей и популярно используются для распознавание лиц и нахождение их координаты в высоком уровне.

**2.4 Нахождение расстояния между камерой и лиц**

Распознаванием лицо наше дело не заканчивается. Библиотека Face Recognition определяет достаточно далеко расположенных лиц. Нам надо задавать порог, который при прохождении этого порога должен открывается замок. Для этого вспомним фундаментальные знания по оптике.



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

В качестве ширины объекта будем брать расстояние между глаз, который составляет для мужчин 6.4см, а для женщин 6.2см. Мы будем брать среднее значение W = 6.3 см.

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

С помощью этого значения мы можем найти расстояние до человека по этой формуле:

**3. Разработка серверной части**

Так мы можем получить изображение, обрабатывать и распознать, но было бы лучшее сохранить эти данные и управлять замками с помощью GUI (Графический интерфейс пользователя). Для этого нам надо разработать серверную часть и мы избежим данный ряд проблем.

1. не понадобятся большие вычислительное мощности в каждом замке, все данные будут обрабатываться в сервере.
2. отслеживание всех устройств в одном месте.
3. удобный интерфейс для конечного пользователя.

Для создания GUI будем использовать веб фреймворк Django в Python, который предоставляет удобный интерфейс, который управляет базой данных и будет работать в локальном хосте.

**3.1 Разработка базу данных в 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 мы можем сосредоточиться на написании своего приложения, не изобретая велосипед. Это бесплатно и с открытым исходным кодом.

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](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) с концепциями объектно-ориентированных языков программирования, создавая «виртуальную [объектную базу](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) данных».

В рамках этой работе нам понадобится создать 3 класса

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

- Имя и Фамилия;

- Идентификационный номер;

- Будет имеет один из трех уровня доступа;

1-уревень - Для гостей и обычных сотрудников;

2-уревень - Для начальства;

3-уревень - Для глав отдела или компании;

- Фото.

1. класс Lock - это класс отвечает за данными замка:

- Идентификационный номер;

- ПОРТ соединения;

- Местоположение;

- Фокальное расстояние (полученное в экспериментально).

1. класс Activity - это класс отвечает за активность:

- Profile (Объект который распознал или не распознал устройство);

- дата и время;

- Lock (через какое устройство прошел).

Таким образом мы составляем базу данных и ими управляем. Остается только соединить эти две подсистемы.

**4.Связь сервера с клиентами**

М**одель взаимодействия клиент-сервер - это** разделение функционал и вычислительную нагрузку между клиентскими и серверными приложениями.



рис.7 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.

**4.1 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.

В рамках нашей работы, сервер открывает определенный порт и будет слушать поступающий запрос, обрабатывать и отвечать на эти запросы. Для увеличения быстродействия мы будем использовать многопоточное программирование. В этом случае каждый запрос независимо друг от друга будет выполняться параллельно.

**5. Управления замком**

Замок управляющей системой пропусков представляет собой 4 компонента:

- микросхема имеющая Ethernet порт для подключения локальному хосту;

- камера для получения кадра;

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

- язычок для блокировки дверей;

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.

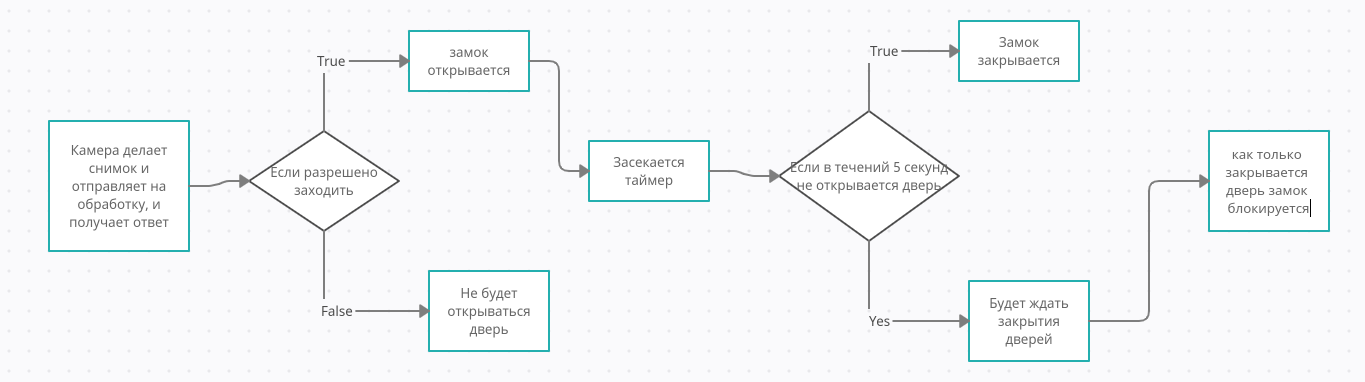


рис.8 Блок схема управления замком

**Заключение**

В данной работе были рассмотрены пошагово разработана пропускная система и были предложены решение следующих задач:

- Алгоритм распознавания лиц;

- Разработка серверного приложения;

- Принцип управления электрическим замком.

Достоинством такой системы является отслеживание в реальном времени и хорошая степень защиты.

**Список использованных источников**

1. Установка систем биометрического распознаванию лиц // <https://ipsdi.ru/biometricheskogo-raspoznavaniju-lic> (дата обращения: 27.01.2022)
2. Как работает распознавание лиц? Разбор // <https://droider.ru/post/kak-rabotaet-raspoznavanie-licz-razbor-07-07-2021/> (дата обращения: 27.01.2022)
3. Официальная сайт OpenCV // <https://opencv.org/> (дата обращения: 28.01.2022)
4. Официальная сайт Face Recognition // <https://github.com/ageitgey/face_recognition> (дата обращения: 28.01.2022)
5. Официальная сайт Django // <https://www.djangoproject.com/> (дата обращения: 28.01.2022)
6. Принцип сокетов // <https://lecturesnet.readthedocs.io/net/low-level/ipc/socket/intro.html> (дата обращения: 28.01.2022)