Министерство науки и высшего образования  
Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Курский государственный университет»

Факультет физики, математики, информатики

Кафедра программного обеспечения и администрирования информационных систем

Курсовой проект

по дисциплине

«Основы проектирования информационных систем и   
баз данных»

на тему: ИНФОРМАЦИОННАЯ СИСТЕМА   
РАСПОЗНАВАНИЯ ЭМОЦИЙ ЧЕЛОВЕКА НА ОСНОВЕ АНАЛИЗА ИЗОБРАЖЕНИЙ

Обучающегося 4 курса

очной формы обучения

направления подготовки

02.03.03 Математическое обеспечение и администрирование информационных систем

Направленность (профиль) Проектирование информационных систем и баз данных

Мазуренко Сергея Викторовича



Руководитель: к.т.н., профессор кафедры ПОиАИС,

Бабкин Евгений Александрович

Допустить к защите:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Бабкин Е.А.

«25» декабря 2020 г.

Курск, 2020

СОДЕРЖАНИЕ

[Введение 3](#_Toc62314364)

[1 Анализ требований к информационной системе 4](#_Toc62314365)

[1.1 Описание и анализ предметной области 4](#_Toc62314366)

[1.2 Обзор и анализ возможных альтернатив 7](#_Toc62314367)

[1.3 Анализ функциональных и эксплуатационных требований 9](#_Toc62314368)

[1.3.1 Стандарты 9](#_Toc62314369)

[1.3.2 Функциональные требования пользователя 10](#_Toc62314370)

[1.3.3 Входные данные 10](#_Toc62314371)

[1.3.4 Выходные данные 11](#_Toc62314372)

[1.3.5 Требования к интерфейсу 11](#_Toc62314373)

[1.3.6 Требования к надежности 11](#_Toc62314374)

[1.3.7 Требования к программной документации 11](#_Toc62314375)

[1.3.8 Требования к составу и параметрам технических средств 12](#_Toc62314376)

[1.3.9 Модель вариантов использования 12](#_Toc62314377)

[1.3.10 Глоссарий проекта 17](#_Toc62314378)

[1.3.11 Проверка модели на полноту 18](#_Toc62314379)

[2 Проектирование информационной системы 21](#_Toc62314380)

[2.1 Разработка архитектуры системы 21](#_Toc62314383)

[2.2 Разработка модели предметной области 22](#_Toc62314384)

[2.3 Разработка алгоритма функционирования системы 23](#_Toc62314385)

[2.4 Проектирование интерфейса пользователя 27](#_Toc62314386)

[2.5 Модель данных 29](#_Toc62314387)

[2.6 Проектирование классов предметной области 30](#_Toc62314388)

[2.6.1 Построение диаграмм последовательности 30](#_Toc62314389)

[2.6.2 Построение диаграммы кооперации 31](#_Toc62314398)

[2.6.3 Построение диаграммы классов 32](#_Toc62314399)

[2.6.4 Уточнение структуры классов предметной области и разработка алгоритмов методов 33](#_Toc62314400)

[3 Реализация системы 35](#_Toc62314401)

[3.1 Реализация программного обеспечения системы 35](#_Toc62314403)

[3.1.1 Разработка диаграммы компонентов 35](#_Toc62314404)

[3.1.2 Объекты интерфейса пользователя 35](#_Toc62314405)

[3.2 Реализация технического обеспечения 40](#_Toc62314406)

[Заключение 42](#_Toc62314407)

[Список использованных источников 43](#_Toc62314408)

[Приложение А Текст фрагмента программы 44](#_Toc62314409)

[Приложение Б Графический материал 52](#_Toc62314410)

**ВВЕДЕНИЕ**

Разрабатываемая информационная система предназначена для реализации распознавания эмоций с использованием веб-камеры и носит название «Информационная система распознавания эмоций по изображению».

Тема курсового проекта актуальна, так как данный программный продукт расширяет возможности декодирования человеческих эмоций на более качественном технологическом уровне. Разрабатываемой продукт может быть полезен структурам правоохранительных органов или для сбора и анализа состояния пользователей для маркетингового исследования, системы аналитики для умных городов, человеко-машинного взаимодействия.

Целью данного курсового проекта является создание информационной системы, позволяющей распознавать эмоции пользователя по изображению с видео.

Основные задачи разработки:

* распознавание лица человека по изображению;
* получение приближенной к объективной оценке эмоционального состояния человека статистики;
* обеспечение отображения информации об эмоциональном состоянии человека на экран;
* обеспечение отображения статистик об общем эмоциональном состоянии человека.

Для реализации проекта был выбран язык программирования C++ и интегрированная среда разработки Microsoft Visual Studio. Для построения моделей используются CASE-средства Rational Rose, CA ERwin Process Modeler и Bizagi Modeler.

1. Анализ требований к информационной системе

Описание и анализ предметной области

В настоящее время самым распространенным способом распознавания эмоций человека, по-прежнему, являются использование такого технического средства как полиграф или, иначе, детектор лжи, для работы которого требуются: датчик для каналов измерения дыхания, датчик для канала фиксации изменения артериального давления (АД), датчик для канала фиксации психологической составляющей (ПС) и другие, что делает такой подход к распознаванию эмоционального состояния человека труднодоступным. Помимо этого, необходимость подключения к человеку датчиков делает процесс распознавания эмоций открытым.

Однако, такой метод нередко подвергается критике. В нем усматривают слишком упрощённую, механическую схему интерпретации реакций человеческого организма на внешние раздражители, кроме того, во время экспертизы могут быть не учтены различные факторы, влияющие на основные показатели.

Сейчас большую популярность набирают бесконтактные способы распознавания эмоций. Область бесконтактного распознавания эмоций включает в себя программные продукты. В списке наиболее популярных программных средств можно выделить Microsoft Oxford Project Emotion Recognition [1], FaceReader [2] компании Noldus, EmoDetect [3] компании Neurobotics. Более подробный анализ программных средств будет производиться в пункте 1.2.

Актуальность предметной области бесконтактного распознавания эмоций велика. Бесконтактное распознавание эмоций предоставляет удобный способ анализа состояния человека.

Полностью обойтись без дополнительного оборудования для данного вида распознавания эмоций нельзя, таким образом для работы с информационной системой понадобится веб-камера. В настоящее время веб-камерой оснащен любой ноутбук, а приобретение данного устройства обладателям настольных компьютеров составит, в среднем, около 30-50 долларов [5] и проблем с её наличием в магазинах нет.

За алгоритм распознавания лиц возьмем алгоритм обнаружения объектов на изображениях в реальном времени, разработанный в 2001 году Полом Виолой и Майклом Джонсом и названный методом Виолы-Джонса [6]. Несмотря на то, что главной задачей при разработке этого метода было нахождение лиц, этот метод используется и для распознавания других различных предметов. Существует множество реализаций, в том числе в составе библиотеки компьютерного зрения OpenCV [5].

Признаки, применяемые алгоритмом, используют суммирование пикселей из прямоугольных регионов. При стандартном размере признака в 24x24 пикселя, возможно 162 тысячи разных признака и производить расчёт каждого слишком долго. Поэтому в алгоритме Виолы-Джонса используется вариация алгоритма обучения AdaBoost [7], как для выбора признаков, так и для настройки классификаторов.

Принцип работы разрабатываемой информационной системы прост, изображение поступает из камеры, которая является либо встроенной веб-камерой ноутбука, либо приобретенной отдельно, в программу, установленную на персональном компьютере пользователя. Все вычисления так же происходят на устройстве пользователя, однако отчеты о работе программы, ее ошибки поступают на удаленный сервер, с целью совершенствования информационной систем и устранения неполадок в ее работе.

На рисунке 1 представлена контекстная DFD-диаграмма, которая позволяют описать функции системы.

На рисунке 2 представлен бизнес-процесс «Отображение статистики» в BPMN.

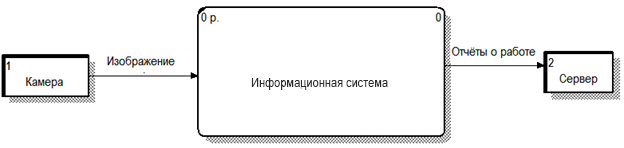


Рисунок 1 – Контекстная DFD-диаграмма разрабатываемой информационной системы

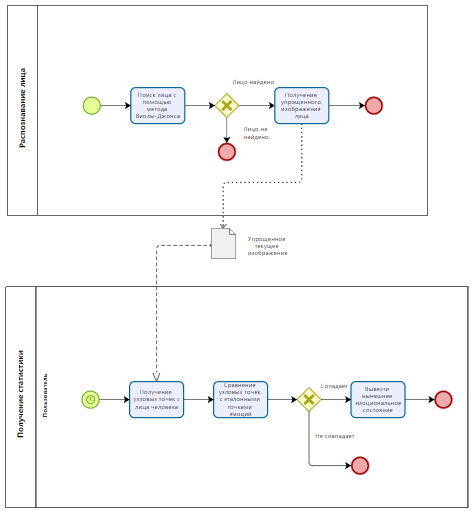


Рисунок 2 – Описание бизнес-процесса «Отображение статистики» в BPMN

Обзор и анализ возможных альтернатив

При поиске альтернативных вариантов разрабатываемого программного продукта были найдены следующие альтернативы:

* Microsoft Oxford Project Emotion Recognition;
* FaceReader;
* EmoDetect.

Машинные алгоритмы перечисленных продуктов анализируют наличие лиц на загружаемой фотографии, после чего определяют по мимике предположительные эмоции. Учитывая тестовый режим работы этого инструмента, результаты часто получаются неточными и неожиданными. Предоставленное API в доступной форме даёт разработчикам всю мощь алгоритма машинного зрения и распознавания эмоций для использования в своих приложениях.

На рисунке 3 представлено главное окно программы Microsoft Oxford Project Emotion Recognition.

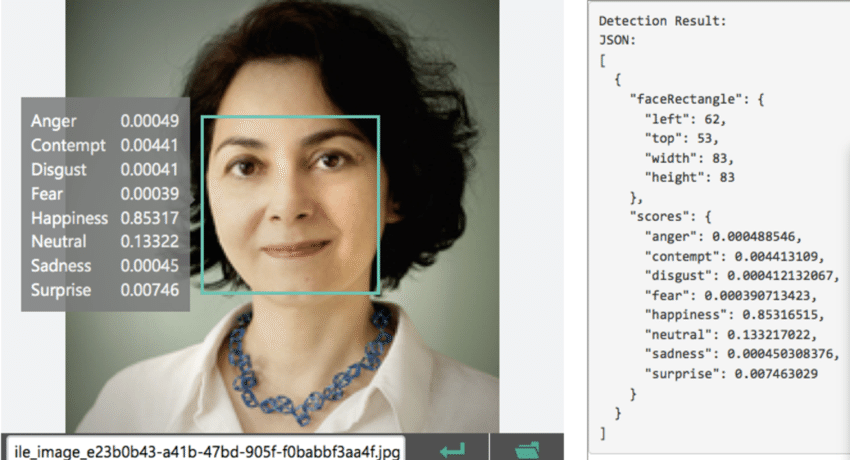


Рисунок 3 – Главное окно программы Microsoft Oxford Project Emotion Recognition

Недостатком программного продукта от Microsoft является анализ эмоций человека по статическому изображению. Все, что требуется пользователю, это загрузить фотографию с изображенным на ней человеком и дождаться результата работы алгоритмов.

На рисунке 4 представлено главное окно программы FaceReader от компании Noldus.

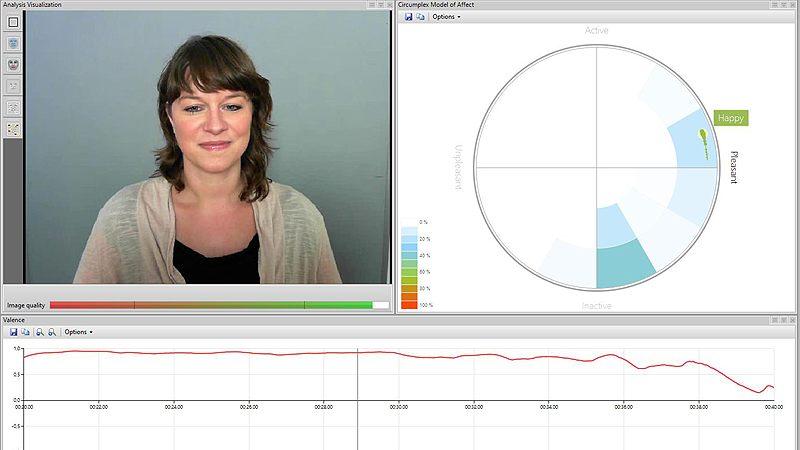


Рисунок 4 – Главное окно программы FaceReader

FaceReader может верно распознавать по выражению лица такие эмоции, как «счастье», «грусть», «злоба», «удивление», «страх», «отвращение» и «спокойствие». Кроме того, приложение способно определить возраст, пол и расу людей. Приложение не нуждается в обучении и дополнительной настройке.

Однако процент распознавания сильно падает, если человек носит очки или голова пользователя повернута в профиль по отношению к камере.

На рисунке 5 представлено главное окно программы EmoDetect от компании Neurobotics.

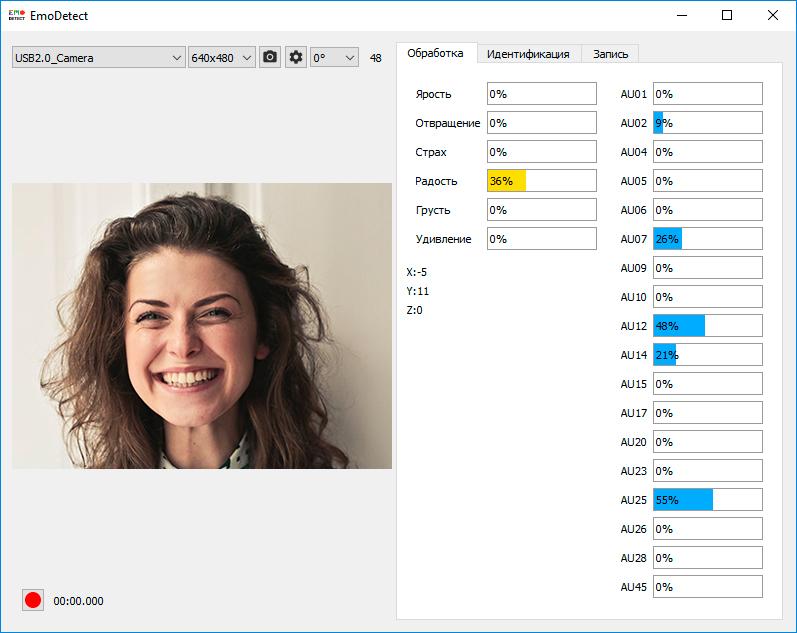


Рисунок 5 – Главное окно программы EmoDetect

Программное обеспечение EmoDetect позволяет определить психоэмоциональное состояние человека по видео или набору файлов. Разработанный классификатор эмоций позволяет определить 6 базовых эмоций: радость, удивление, грусть, злость, страх, отвращение. Определение эмоций происходит с помощью нейронных сетей: выполняется нахождение опорных точек на лице человека, и классификатор строит общую картину на основе их положения относительно друг друга.

Недостатком данного продукта является то, что программа не распознает эмоции человека, повернутого в профиль камере.

Анализ функциональных и эксплуатационных требований

Стандарты

Программный продукт разрабатывается на основании следующих государственных стандартов:

1. Межгосударственный стандарт ГОСТ 7.32-2018 «Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно – исследовательской работе. Структура и правила оформления».

2. Международный стандарт ISO/IEC 12207. Информационные технологии. Процессы жизненного цикла программного обеспечения.

3. ГОСТ 34.601-90. Автоматизированные системы. Стадии создания.

4. ГОСТ 34.602-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы.

5. ГОСТ 34.603-92. Информационная технология. Виды испытаний автоматизированных систем.

Функциональные требования пользователя

Программный продукт, разрабатываемый в рамках курсового проекта, должен удовлетворять следующему перечню функциональных требований (рис. Б.1):

* обеспечение автоматизированного входа пользователей в систему (по соответствующему пункту настроек);
* установка фокуса анализа на определённого (временного) пользователя (если в кадре находится несколько человек);
* сбор и анализ полученного с пользователя эмоционального состояния.

Входные данные

Входными данными при работе с программой являются изображения, поступающие с веб-камеры в режиме реального времени.

Выходные данные

Выходными данными при работе программы является общая статистика о состоянии человека и статистика в данный момент времени, выводимая в самом клиентском приложении и является частью его интерфейса.

Требования к интерфейсу

У пользователя должна быть возможность войти в систему, или же, если пользователь не авторизирован, возможность зарегистрироваться.

У программного продукта должна присутствовать доступная пользователю возможность настройки качества изображения веб-камеры.

Кроме того, должна быть возможность увидеть статистику по своему эмоциональному состоянию.

Требования к надежности

При работе с программным продуктом необходимо предусмотреть:

* блокировку некорректных действий пользователя при работе с системой;
* разграничение доступа пользователей.

Требования к программной документации

В состав сопровождающей документации программного продукта должны входить следующие компоненты:

1. Пояснительная записка на 30 – 50 листах, содержащая описание разработки и исходные тексты основных модулей на языке C++.

2. Пояснительная записка, исходные тексты модулей на языке C++, задание на курсовой проект и справка о прохождении антиплагиата на CD-диске.

Требования к составу и параметрам технических средств

Для работы программы желательно иметь IBM-совместимый персональный компьютер со следующей минимальной характеристикой:

* Двухъядерный процессор с тактовой частотой 2 ГГц;
* объём оперативного запоминающего устройства – 2 Гб;
* операционная система – Windows 7 – 10 x64;
* объём свободного места на жёстком диске - 128 Мб;
* видеоадаптер SVGA с цветным дисплеем;
* веб-камера.

Модель вариантов использования

Действующие лица для диаграммы вариантов использования приведены в таблице 1.

На основании анализа требований пользователя были выделены следующие варианты использования, представленные в таблице 2.

На основании всех выше рассмотренных вариантов использования была составлена диаграмма вариантов использования, представленная на рисунке 6.

Таблица 1 – Действующие лица

|  |  |
| --- | --- |
| **Термин** | **Значение** |
| Пользователь | Пользователь информационной системой |
| Сервер | Сервер, предназначенный для сбора информации о работе программы |

Таблица 2 – Описание вариантов использования

|  |  |
| --- | --- |
| **Термин** | **Значение** |
| Получение статистики | Просмотр статистики об эмоциональном состоянии. Статистика может быть как подробной, за все время использование программы, так и краткой, т.е. в текущий момент времени |
| Ознакомление с руководством пользования | Просмотр инструкции по использованию программы пользователем |
| Регистрация пользователя в системе | Создание учётных записей пользователей информационной системы |
| Вход в учётную запись пользователя | Автоматизированный вход в учетную запись зарегистрированного пользователя информационной системы |
| Отправка отчёта о работе программы | Отправка отчёта о работе программы через сеть Интернет |
| Изменение настроек | Изменение настроек информационной системы пользователем |

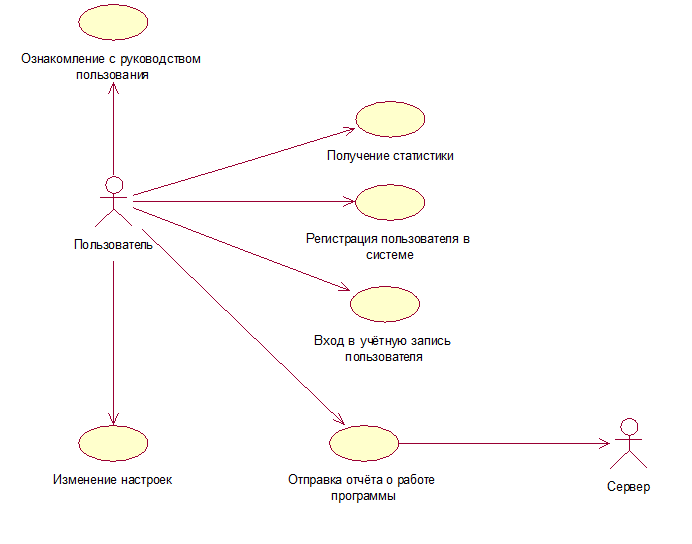


Рисунок 6 – Диаграмма вариантов использования

Рассмотрим описание варианта использования **«**Получение статистики»

Действующие лица:пользователь.Заинтересованные лица и их требования: пользователь хочет посмотреть статистику.

Сценарий варианта использования **«**Получение статистики»представлен в таблице 3.

Таблица 3 – Сценарий **«**Получение статистики»

|  |  |
| --- | --- |
| **Действия пользователя** | **Действия (Отклик) системы** |
| 1. Пользователь запускает программу | 2. Система получает изображение с веб-камеры. |
|  | 3. Система выполняет поиск лица пользователя на изображении. |
| Альтернативный поток событий А1. Пользователь не найден. | |
| Альтернативный поток событий А2. На изображении несколько лиц. | |

Продолжение таблицы 3

|  |  |
| --- | --- |
| **Действия пользователя** | **Действия (Отклик) системы** |
|  | 4. Система вычисляет и анализирует информацию, полученную с лица пользователя. |
|  | 5. Система отображает статистику. |

Альтернативный поток событий А1. Пользователь не найден: если лицо пользователя, осуществляющего управление не найдено на изображении, то система оповещает об этом пользователя.

Альтернативный поток событий А2. На изображении несколько лиц: если на изображении присутствуют несколько лиц, то система должна реагировать на лицо текущего пользователя.

Предусловия: пользователь вошёл в систему.

Постусловия: если вариант использования выполнен успешно, система отображает статистику об эмоциональном состоянии пользователя.

Рассмотрим описание варианта использования «Регистрация пользователя в системе**»**

Действующие лица: пользователь. Заинтересованные лица и их требования: текущий пользователь хочет создавать учётные записи для новых пользователей информационной системы.

Сценарий варианта использования «Регистрация пользователя в системе**»** представлен в таблице 4.

Таблица 4 – Сценарий «Регистрация пользователя в системе**»**

|  |  |
| --- | --- |
| **Действия пользователя** | **Действия (Отклик) системы** |
| 1. Пользователь вводит данные о новом пользователе. | 2. Система получает данные о пользователе. |

Продолжение таблицы 4

|  |  |
| --- | --- |
| **Действия пользователя** | **Действия (Отклик) системы** |
|  | 3. Система получает изображение с камеры. |
| Альтернативный поток событий А1. Лицо не найдено. | |
| Альтернативный поток событий А2. Найдено несколько лиц. | |
|  | 4. Система сохраняет информацию о новом пользователе. |

Альтернативный поток событий А1. Лицо не найдено: если на изображении нет лиц, то система возвращается в исходное состояние.

Альтернативный поток событий А2. Найдено несколько лиц: если на изображении несколько лиц, то система предлагает выбрать нового пользователя из доступных на изображении.

Предусловия: пользователь вошёл в систему.

Постусловия: если вариант использования выполнен успешно, то создаётся учётная запись нового пользователя информационной системы.

Рассмотрим описание варианта использования «Вход в учётную запись пользователя»

Действующие лица: пользователь. Заинтересованные лица и их требования: пользователь хочет, чтобы система обеспечивала вход в учётную запись по фотографии, полученной с веб-камеры.

Сценарий варианта использования «Вход в учётную запись пользователя» представлен в таблице 5.

Таблица 5 – Сценарий «Вход в учётную запись пользователя»

|  |  |
| --- | --- |
| **Действия пользователя** | **Действия (Отклик) системы** |
|  | 1. Система получает изображение с камеры. |

Продолжение таблицы 5

|  |  |
| --- | --- |
| **Действия пользователя** | **Действия (Отклик) системы** |
|  | 2. Система обрабатывает изображение. |
| Альтернативный поток событий А1. Лицо не найдено. | |
| Альтернативный поток событий А2. На изображении несколько лиц. | |
| Альтернативный поток событий А3. Пользователь не найден. | |
|  | 3. Выполняется вход в учетную запись пользователя. |

Альтернативный поток событий А1. Лицо не найдено: если на изображении нет лиц, то система возвращается в исходное состояние.

Альтернативный поток событий А2. На изображении несколько лиц: если на изображении несколько лиц, то система проводит анализ лица, выбранного пользователем.

Альтернативный поток событий А3. Пользователь не найден: если данные о лице пользователя не совпадают с имеющимися, то система возвращается в исходное состояние.

Предусловия: список пользователей не пуст и выполняется событие смены пользователя.

Постусловия: если вариант использования выполнен успешно, то осуществляется вход в учётную запись пользователя.

Глоссарий проекта

В таблице 6 приведены термины предметной области и их значения.

Таблица 6 – Термины и их значения

|  |  |
| --- | --- |
| **Термин** | **Значение** |
| Пользователь | Человек, который работает с информационной системой |
| Учётная запись | Совокупность данных о пользователе информационной системы |
| Отчёт о работе | Информация о работе программного продукта на персональном компьютере пользователя |

Проверка модели на полноту

Проверка на полноту диаграммы вариантов использования производится по операциям, выполняемым над основными объектами. Результат проверки представлен в таблице 7.

Основными объектами, упоминаемыми в модели вариантов использования, являются: учётная запись, руководство пользования, настройки, отчёт и статистика.

В таблице 7 обозначены виды операций:

* 1. просмотр;
  2. создание, генерация;
  3. отправка;
  4. вход;
  5. изменение;

Операции 1-5 производятся над определёнными объектами, соответствующими разным состояниям информационной системы. Все операции над объектами выполняются пользователем, кроме операции (1) объекта «Статистика», операции (2) объекта «Отчёт» и объекта «Статистика» и операции (4) объекта «Учётная запись». Данные операции выполняются информационной системой автоматически без действий пользователя.

Таблица 7 - Проверка на полноту

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Варианты использования** | **Объекты** | | | | |
| Статистика | Учётная запись | Руководство пользования | Настройки | Отчёт |
| Отображение статистики | 1, 2 |  |  |  |  |
| Ознакомление с руководством пользования |  |  | 1 |  |  |
| Регистрация пользователя в системе |  | 2 |  |  |  |
| Вход в учётную запись пользователя |  | 4 |  |  |  |
| Отправка отчёта о работе программы |  |  |  |  | 2, 3 |
| Изменение настроек |  |  |  | 5, 1 |  |

Результаты анализа полноты выполнения функциональных требований пользователя в модели вариантов использования приведены в таблице 8. Все функциональные требования пользователя отражены в основных вариантах использования.

Таблица 8 - Анализ полноты выполнения требований пользователя

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Требования пользователя** | **Варианты использования** | | | | | |
| Отображение статистики | Ознакомление с руководством пользования | Регистрация пользователя в системе | Вход в учётную запись пользователя | Отправка отчёта о работе программы | Изменение настроек |
| Разграничение доступа пользователей и защита данных | + |  |  | + |  |  |
| Отображение статистики | + |  |  |  |  |  |
| Просмотр руководства пользования |  | + |  |  |  |  |
| Создание новых учётных записей пользователей |  |  | + |  |  |  |
| Изменение настроек работы программы |  |  |  |  |  | + |
| Генерация отчётов о работе информационной системы и отправка по разрешению пользователя |  |  |  |  | + |  |

1. Проектирование информационной системы
3. 1. Разработка архитектуры системы

Для работы пользователя с информационной системой потребуется наличие персонального компьютера, с подключенной к нему веб-камерой и доступом к Интернету, для отправки отчетов. В процессе работы приложения, по разрешению пользователя, генерируются отчёты о работе программного обеспечения, которые отправляются на сервер.

Предварительная диаграмма развёртывания представлена на рисунке 7.

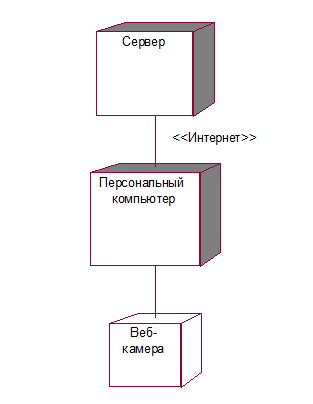


Рисунок 7 ­­– Архитектура технических средств системы

* 1. Разработка модели предметной области

Модель предметной области включает в себя набор классов, представляющих последовательность обработки изображения, получаемого с веб-камеры, установленной на персональном компьютере пользователя.

Список концептуальных классов:

* Камера;
* Пользователь;
* Статистика.

На основании анализа словесного описания варианта использования, составлен список ассоциаций для предметной области, представленный в таблице 9.

Таблица 9 - Ассоциации для модели предметной области

|  |  |
| --- | --- |
| **Ассоциация** | **Описание ассоциации** |
| Распознаёт пользователя | Распознавание лица пользователя и его передача модулю обработки изображения |
| Получение статистики | Обработка изображения с лицом пользователя и генерация статистики |

На основании анализа технического задания и описания вариантов использования выделены атрибуты классов для модели предметной области, представленные в таблице 10.

Таблица 10 - Атрибуты классов для модели предметной области

|  |  |
| --- | --- |
| **Название класса** | **Атрибуты класса** |
| Камера | instance – экземпляр объекта класса Камера |

Продолжение таблицы 10

|  |  |
| --- | --- |
| **Название класса** | **Атрибуты класса** |
| Пользователь | instance – экземпляр объекта класса Пользователь  Имя пользователя  Инициалы пользователя  Данные об изображении  Учетная запись в ОС |
| Статистика | instance – экземпляр объекта класса Статистика |

Концептуальная модель предметной области представлена на рисунке 8 и рисунке Б.2 приложения Б.

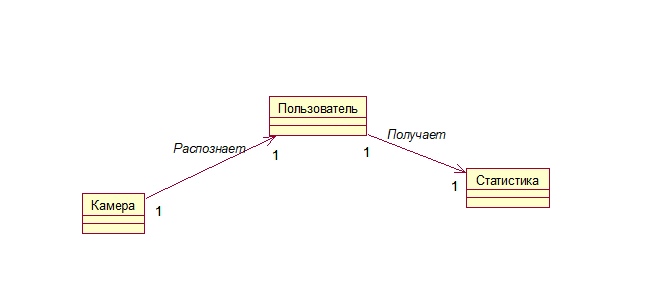


Рисунок 8 - Концептуальная модель предметной области

* 1. Разработка алгоритма функционирования системы

В разрабатываемом программном продукте предусмотрена подсистема разграничения доступа пользователей к их учётным записям. Пользователю, запустившему программу впервые, требуется создать собственную учётную запись, которая привязывается к локальной учётной записи пользователя операционной системы. Если пользователь создавал учётную запись ранее, то ему необходимо войти в неё и после этого создать учётные записи для новых пользователей.

Алгоритм работы подсистемы регистрации новых пользователей представлен на рисунке 9.

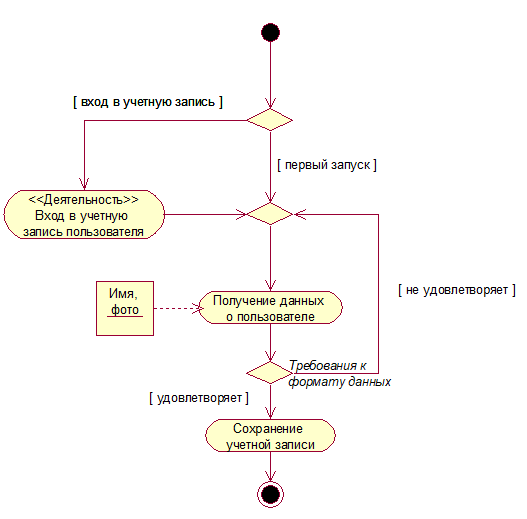


Рисунок 9 - Диаграмма деятельностей «Регистрация пользователя в системе»

При входе в учётную запись пользователя, анализируется изображение, полученное с веб-камеры и данные об изображениях с лицами зарегистрированных в системе пользователей.

Алгоритм входа в учётную запись пользователя представлен на

рисунке 10.

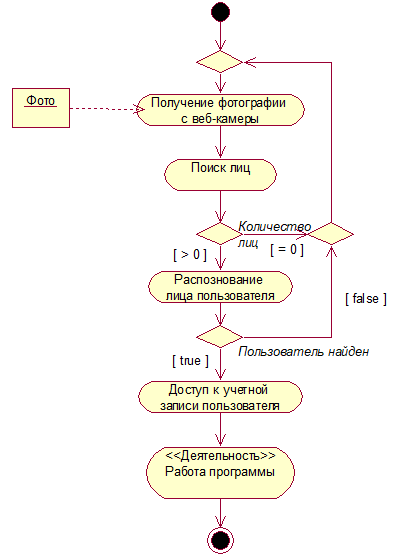


Рисунок 10 - Диаграмма деятельностей «Вход в учётную запись пользователя»

После входа в учётную запись пользователя происходит автоматический запуск программы управления компьютером. Система производит поиск лица текущего пользователя и его обработку программы.

Алгоритм работы программы представлен на рисунке 11.

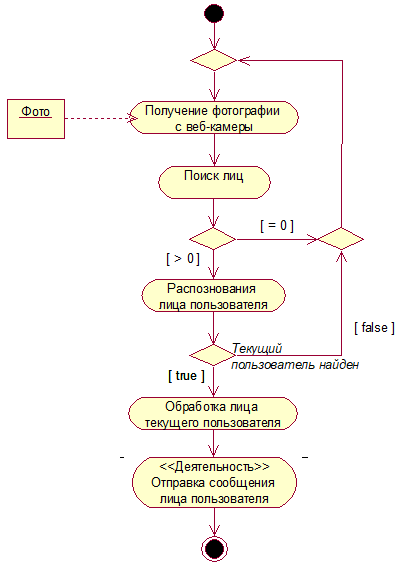


Рисунок 11 - Диаграмма деятельностей «Работа программы»

Во время работы программы доступна возможность переключения фокуса управления с текущего на временного пользователя.

На рисунке 12 представлена деятельность пользователя «Переключение фокуса на временного пользователя».

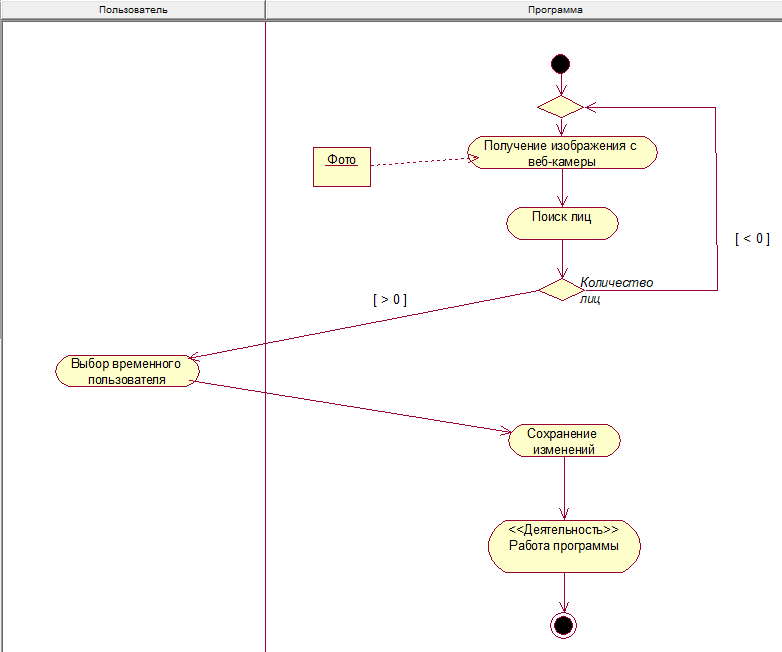


Рисунок 12 - Диаграмма деятельностей «Переключение фокуса на временного пользователя»

* 1. Проектирование интерфейса пользователя

На основании алгоритмов функционирования информационной системы и требований к интерфейсу была разработана диаграмма состояний пользовательского интерфейса.

Диаграмма состояний пользовательского интерфейса представлена на рисунке 13 и рисунке Б.3 приложения Б.

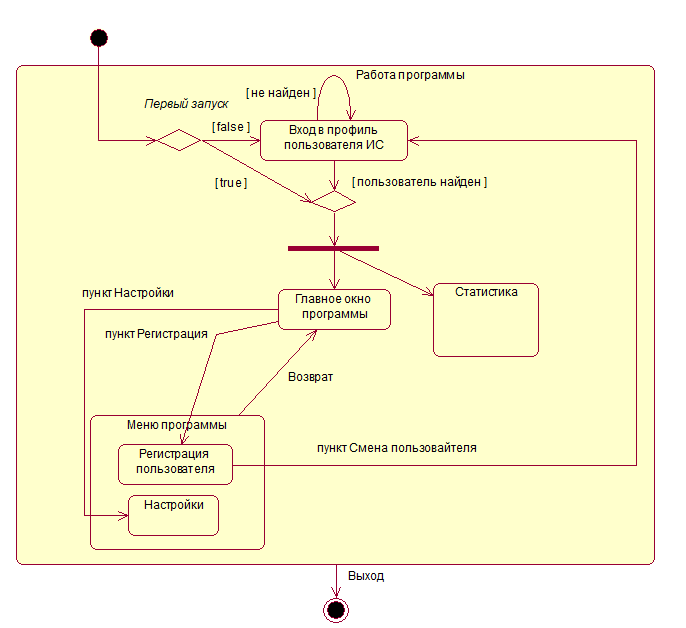


Рисунок 13 - Диаграмма состояний пользовательского интерфейса

При запуске приложения определяется было ли оно запущено впервые или нет. Если программный продукт запускается впервые, то осуществляется переход на главное окно. Фокус управления переключается на пользователя, расположенного ближе к веб-камере. Считается, что программа запущена впервые до тех пор, пока в системе нет профилей пользователей. Если в информационной системе имеются профили пользователей, то осуществляется переход в окно входа в систему.

При успешном входе пользователя в систему происходит открытие главного окна. Из меню главного окна имеется доступ к окнам регистрации пользователей и настроек. Суперсостояние интерфейса «Меню программы» показывает, что из окон регистрации пользователей и настроек осуществляется возврат на главное окно. Из окна регистрации можно осуществить смену учётной записи пользователя.

При входе в систему, также происходит инициализация окна статистики, доступ к которой должен быть осуществлён из любого запущенного приложения.

Суперсостояние «Управление» указывает на то, что при закрытии приложения осуществляется закрытие всех окон пользовательского интерфейса программы, в том числе и статистики.

* 1. Модель данных

Разрабатываемая информационная система содержит данные об учётных записях пользователя, которые хранятся в отдельном файле на персональном компьютере. Структура записи информации о зарегистрированном пользователе представлена на рисунке 14.

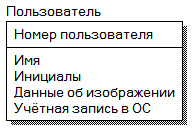


Рисунок 14 – Структура записи зарегистрированного пользователя

* 1. Проектирование классов предметной области
     1. Построение диаграмм последовательности

Руководствуясь описаниями варианта использования «Распознавание эмоций» и спецификой программной реализации, выделен поток, осуществляющий распознавание эмоций.

Диаграмма последовательности, описывающая поток, осуществляющий распознавание эмоций представлена на рисунке 15 и рисунке Б.4

приложения Б.

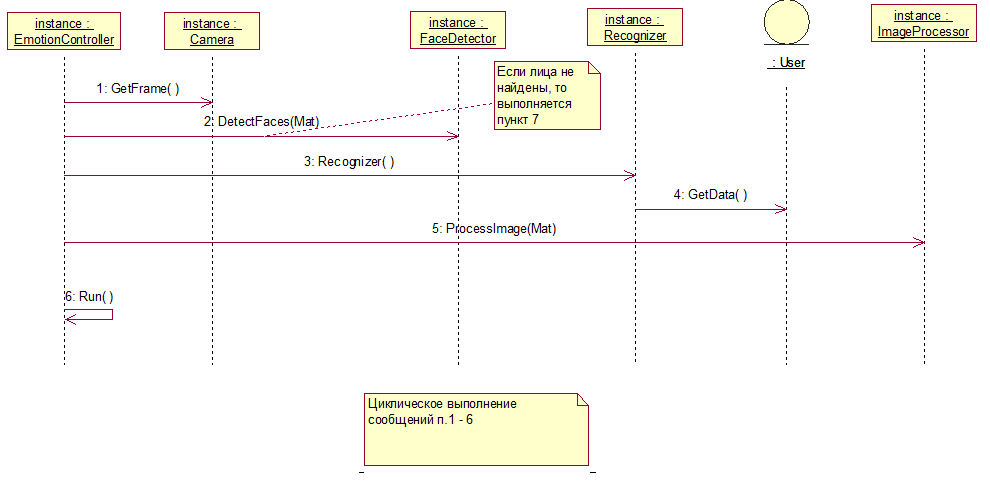


Рисунок 15 – Диаграмма последовательности, описывающая поток, осуществляющий распознавание эмоций



### Построение диаграммы кооперации

Структурные особенности передачи и приёма сообщений между объектами потока, осуществляющего распознавание эмоций представлены в диаграмме кооперации на рисунке 16.

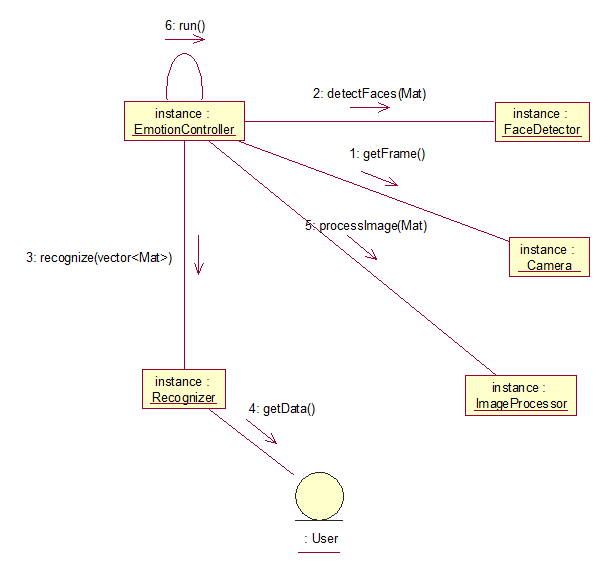


Рисунок 16 – Диаграмма кооперации для потока, осуществляющего распознавание эмоций

### Построение диаграммы классов

Полная диаграмма классов разрабатываемой системы представлена на рисунке 17 и рисунке Б.5 приложения Б.

На приведённой диаграмме приведены основные классы, представляющие логику информационной системы.

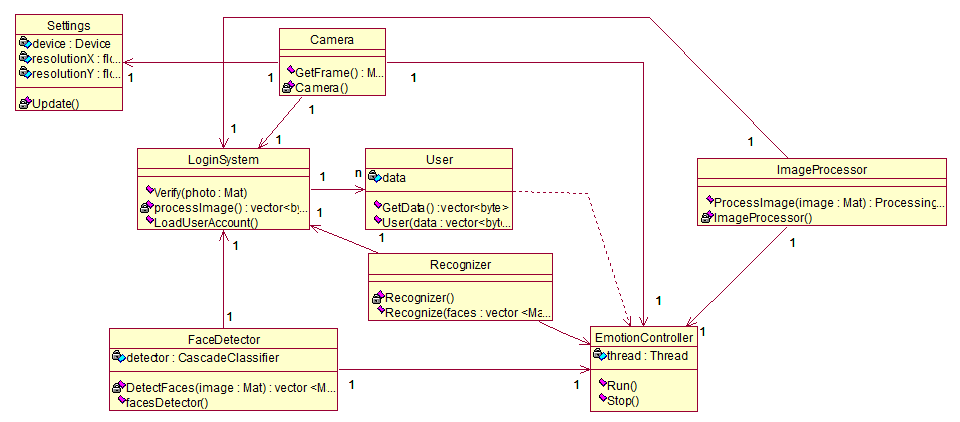


Рисунок 17 – Полная диаграмма классов разрабатываемой системы

Классы FaceDetector, Recognizer и ImageProcessor содержат методы, для работы с изображениями в режиме реального времени, а именно:

* класс FaceDetector предназначен для осуществления поиска лиц на изображении;
* класс Recognizer распознает лицо текущего пользователя, данные о котором хранятся в объекте класса User;
* класс ImageProcessor предназначен для обработки изображения лица текущего пользователя. В нём происходит распознавание эмоций человека.

Класс Camera предоставляет метод для получения кадра с веб-камеры в режиме реального времени.

Классы LoginSystem и EmotionController для получения и обработки изображения используют классы Camera, FaceDetector, Recognizer и ImageProcessor.

Класс LoginSystem содержит набор методов, обеспечивающих регистрацию пользователя, либо вход в учётную запись, если такая имеется.

Класс User содержит данные о текущем пользователе. Класс LoginSystem изменяет состояние текущего пользователя, а работа класса EmotionController зависит от состояния объекта класса User.

Класс Settings содержит поля, содержащие значения пользовательских настроек приложения.

### Уточнение структуры классов предметной области и разработка алгоритмов методов

Класс Camera осуществляет работу с веб-камерой. Методы класса представлены в таблице 11.

Класс FaceDetector производит поиск лиц на изображении. Атрибуты класса представлены в таблице 12. Методы класса представлены в таблице 13.

Класс Recognizer производит распознавание лица пользователя на изображении. Методы класса представлены в таблице 14.

Класс ImageProcessor выполняет обработку изображения лица для распознавания эмоций. Методы класса представлены в таблице 15.

Таблица 11 – Методы класса Camera

|  |  |
| --- | --- |
| **Операция** | **Описание** |
| Camera() | Конструктор |
| GetFrame() | Получает кадр с веб-камеры. Возвращает изображение, представленное в виде двумерной матрицы – объект класса Mat |

Таблица 12 - Атрибуты класса FaceDetector

|  |  |  |
| --- | --- | --- |
| **Атрибут** | **Тип** | **Описание** |
| detector | CascadeClassifier | Вспомогательный объект, предназначенный для обнаружения лиц |

Таблица 13 – Методы класса FaceDetector

|  |  |
| --- | --- |
| **Операция** | **Описание** |
| FacesDetector () | Конструктор |
| detectFaces(Mat image) | Выполняет обнаружение лиц. Возвращает динамический массив с изображениями обнаруженных лиц, представленных объектами класса Mat |

Таблица 14 – Методы класса Recognizer

|  |  |
| --- | --- |
| **Операция** | **Описание** |
| Recognizer () | Конструктор |
| recognizer(vector<Mat> faces) | Выполняет распознавание лица текущего пользователя. Возвращает изображение лица текущего пользователя, представленное объектом класса Mat |

Таблица 15 – Методы класса ImageProcessor

|  |  |
| --- | --- |
| **Операция** | **Описание** |
| ImageProcessor () | Конструктор |
| processImage(Mat image) | Выполняет обработку изображения лица с помощью нейронной сети. Возвращает данные об обработке изображения – объект класса ProcessingData |

1. Реализация системы

## Реализация программного обеспечения системы

Разработка диаграммы компонентов

Реализация программного обеспечения системы представлена на рисунке 17 в виде диаграммы компонентов. Она определяет архитектуру разрабатываемой системы на физическом уровне и представляет зависимости между программными компонентами. Исходный код программы приведён в приложении А. Разработка производилась в среде IDE Microsoft Visual Studio 2019 на языке программирования C++ [9].

Объекты интерфейса пользователя

Система включает в себя следующий список форм:

* MainWindow – главное окно программы. Предназначено для вывода информации о работе программы;
* SettingsWindow – окно настроек програмы. Предназначено для изменения настроек работы программы;
* SigninWindow – окно регистрации нового пользователя. Предназначено для создания учётных записей новых пользователей и смены текущего пользователя;
* LoginWindow – окно входа в учётную запись пользователя. Предназначено для разграничения доступа пользователей к персональному компьютеру.

Диаграмма компонентов представлена на рисунке 18.

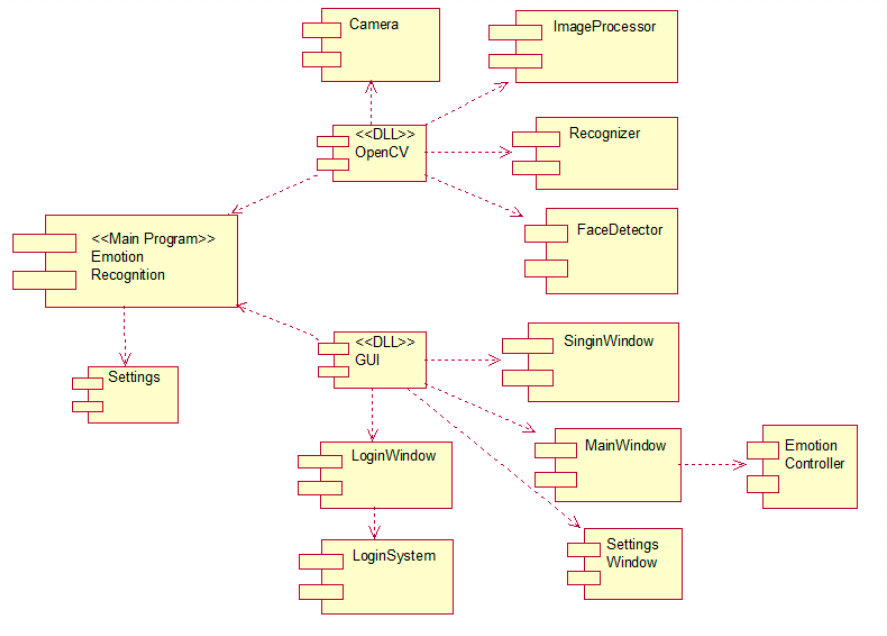


Рисунок 18 – Диаграмма компонентов

Программный продукт включает в свой состав следующий список окон: MainWindow, SettingsWindow, SignInWindow и LogInWindow.

Внешний вид главного окна MainWindow приложения представлен на рисунке 19.

В таблице 16 приведён список компонентов главного окна.

Таблица 16 - Компоненты главного окна

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название компонента** | **Тип компонента** | **Назначение** |
| 1 | settingsMenuItem | ToolStripMenuItem | Открытие окна настроек программы |
| 2 | accountMenuItem | ToolStripMenuItem | Открытие окна создания новых учётных записей |
| 3 | cameraItem | Image | Компонент, выводящий изображение с камеры |
| 4 | statisticLabel | Label | Отображение надписи «Общая статистика» |

Продолжение таблицы 16

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название компонента** | **Тип компонента** | **Назначение** |
| 5 | statisticChart | Chart | Отображение графика общего эмоционального состояния |
| 6 | stateView | Label | Отображение текущего эмоционального состояния |
| 7 | stateLabel | Label | Отображение надписи «Состояние» |

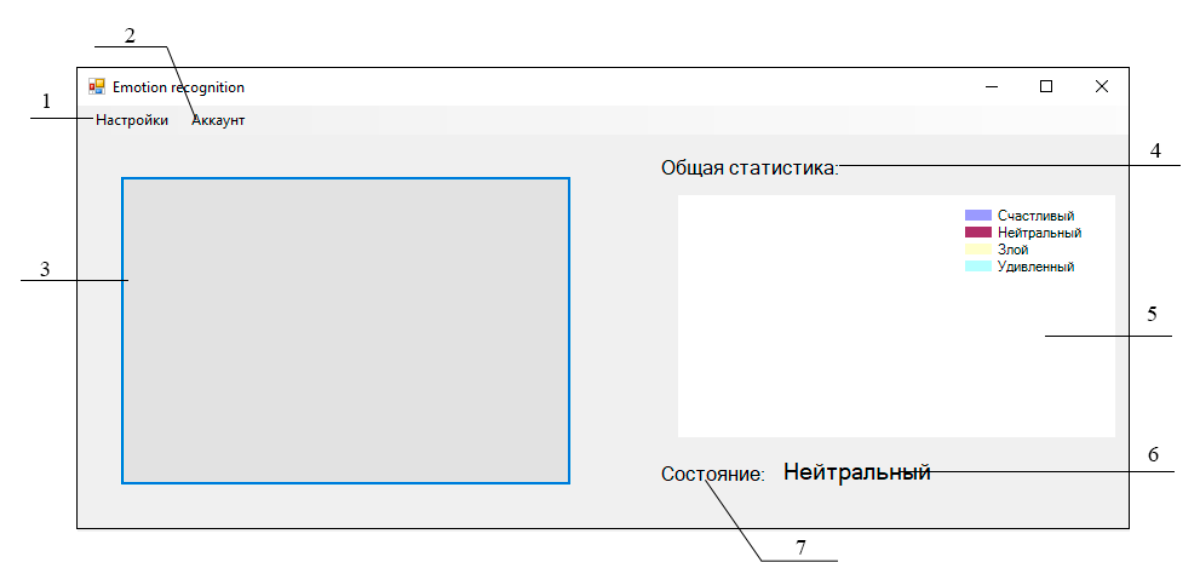


Рисунок 19 – Внешний вид главного окна

Внешний вид окна SettingsWindow настроек приложения представлен на рисунке 20.

В таблице 17 приведён список компонентов окна настроек.

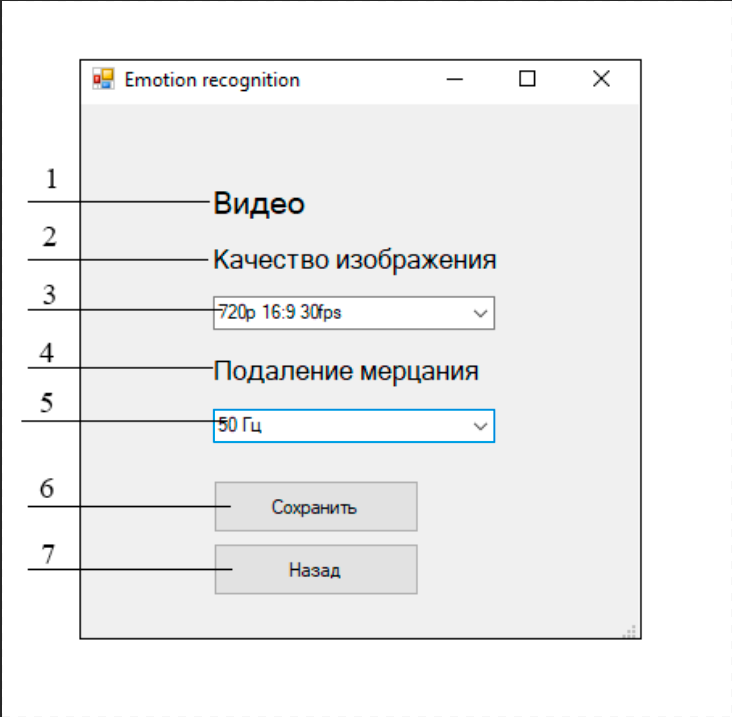


Рисунок 20 – Внешний вид окна настроек

Таблица 17 - Компоненты окна настроек

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название компонента** | **Тип компонента** | **Назначение** |
| 1 | Label1 | Label | Отображение надписи «Видео» |
| 2 | Label2 | Label | Отображение надписи «Качество изображения» |
| 3 | qualityComboBox | ComboBox | Список разрешений качества камеры |
| 4 | Label3 | Label | Отображение надписи «Подавление мерцания» |
| 5 | flickerComboBox | ComboBox | Список доступных Гц для подавления мерцания |
| 6 | saveButton | Button | Кнопка сохранения настроек |
| 7 | closeButton | Button | Кнопка выхода из окна настроек |

Внешний вид окна SignInWindow создания новых учётных записей пользователей приложения представлен на рисунке 21.

В таблице 18 приведён список компонентов окна создания новых учётных записей пользователей приложения.

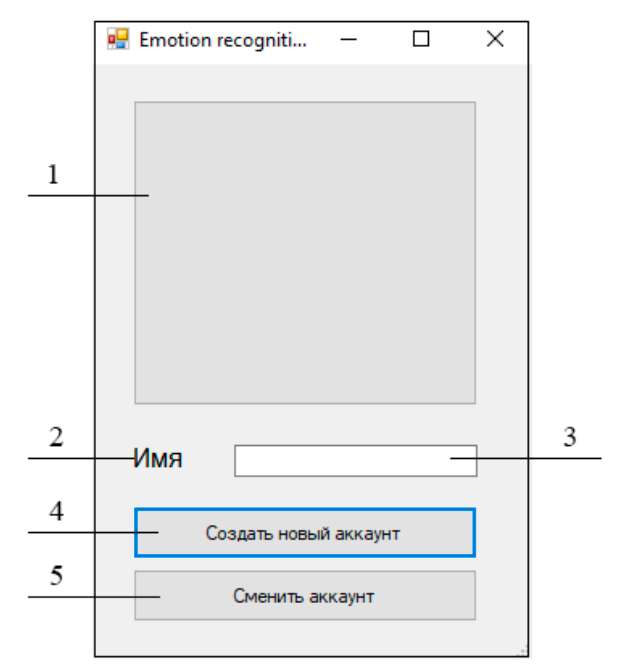


Рисунок 21 – Внешний вид окна создания новых учётных записей пользователей приложения

Таблица 18 - Компоненты окна настроек

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название компонента** | **Тип компонента** | **Назначение** |
| 1 | cameraItem | Image | Компонент, выводящий изображение с камеры |
| 2 | nameLabel | Label | Отображение надписи «Имя» |
| 3 | nameBox | TextBox | Ввод имени нового пользователя |
| 4 | createButton | Button | Кнопка создания нового аккаунта |
| 5 | changeButton | Button | Кнопка смены аккаунта |

Внешний вид окна LogInWindow входа в учётную запись пользователя приложения представлен на рисунке 22.

В таблице 19 приведён список компонентов окна входа в учётную запись пользователя.

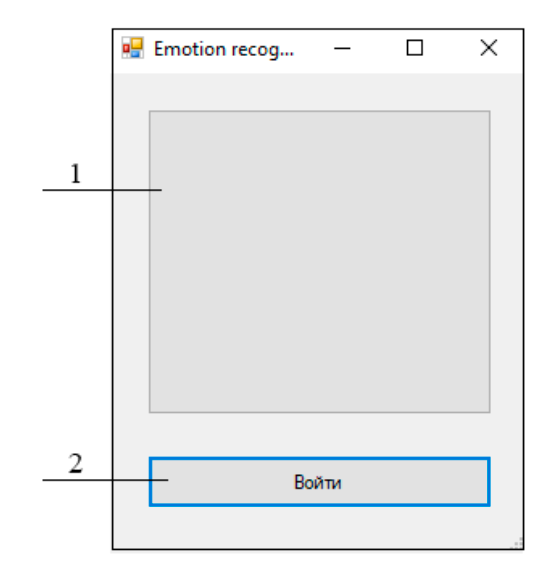


Рисунок 22 – Внешний вид окна входа в учётную запись пользователя

Таблица 19 - Компоненты окна входа в учётную запись пользователя

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название компонента** | **Тип компонента** | **Назначение** |
| 1 | cameraItem | Image | Вывод изображения с веб-камеры |
| 2 | loginButton | Button | Кнопка входа в учётную запись пользователя |

## 

## Реализация технического обеспечения

Для работы с информационной системой пользователю понадобится персональный компьютер с веб-камерой, на котором размещён исполняемый файл – Emotion Recognition.exe. Программа генерирует отчёты о работе, для отправки которых понадобится разрешение пользователя и подключение к сети Интернет. Отправка отчётов осуществляется на сервер с установленной системой обработки отчётов о работе программы на компьютерах клиентов.

Полная диаграмма развертывания системы представлена на рисунке 23.

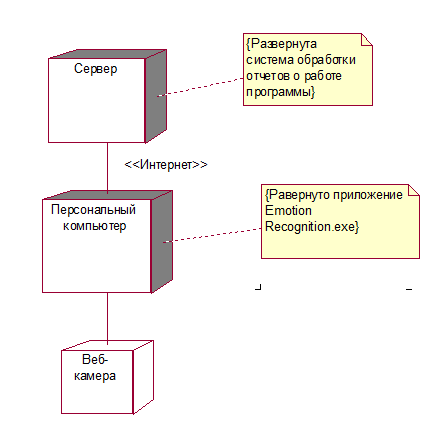


Рисунок 23 – Диаграмма развертывания системы

Заключение

В ходе выполнения данного курсового проекта были выполнены:

* анализ возможных альтернатив (программ для распознавания эмоционального состояния по изображению);
* анализ программных средств для разработки информационной системы;
* анализ библиотек компьютерного зрения;
* анализ методов поиска лиц на изображении;
* разработка архитектуры системы;
* проектировка графического интерфейса пользователя.

Программная реализация информационной системы включает в себя модуль взаимодействия с веб-камерой, библиотеку компьютерного зрения OpenCV 4.5.1 [5] и графический интерфейс пользователя. Задачи, выполненные в рамках курсового проекта, полностью удовлетворяют поставленным.

Информационная система обладает следующим функционалом:

* подробная характеристика эмоционального состояния пользователя;
* изменение настроек работы программы;
* взаимодействие с веб-камерой;
* поиск лиц на изображении, полученном с веб-камеры.

Перечисленный функционал информационной системы был полностью протестирован и отлажен.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Happy? Sad? Angry? This Microsoft tool recognizes emotions in pictures: [Электронный ресурс] URL: https://blogs.microsoft.com/ai/happy-sad-angry-this-microsoft-tool-recognizes-emotions-in-pictures (дата обращения 19.10.2020)
2. Официальный сайт проекта FaceReader: [Электронный ресурс] URL: https://www.noldus.com/facereader (дата обращения 21.10.2020)
3. Официальный сайт проекта EmoDetect: [Электронный ресурс] URL: https://emodetect.ru (дата обращения 22.10.2020)
4. Информация о стоимости веб-камер [Электронный ресурс] URL: https://market.yandex.ru/catalog--veb-kamery/55350 (дата обращения 26.10.2020)
5. Библиотека OpenCV 4.5.1: [Электронный ресурс] URL: https://staging.opencv.org (дата обращения 21.11.2020)
6. Метод Виолы-Джонса (Viola-Jones) как основа для распознавания лиц: [Электронный ресурс] URL: https://habr.com/ru/post/133826/ (дата обращения 21.11.2020)
7. Алгоритм AdaBoost: [Электронный ресурс] UML: http://www.machinelearning.ru/wiki/index.php?title=AdaBoost (дата обращения 21.11.2020)
8. Понимание алгоритма Фурье: [Электронный ресурс] UML: https://habr.com/ru/company/otus/blog/449996/ (дата обращения 21.11.2020)
9. Документация по Microsoft C++: [Электронный ресурс] UML: https://docs.microsoft.com/ru-ru/cpp/?view=msvc-160 (дата обращения 23.11.2020)
10. Руководство по языку программирования C++: [Электронный ресурс] UML: https://metanit.com/cpp/tutorial/ (дата обращения 25.11.2020)

Приложение А

**Текст фрагмента программы**

Приводится фрагмент программы из файла FaceDetector.cpp модуля OpenCV.dll

#include <iostream>

using namespace [std](https://docs.opencv.org/4.5.1/d8/dcc/namespacestd.html);

using namespace [cv](https://docs.opencv.org/4.5.1/d2/d75/namespacecv.html);

static void help(const char\*\* argv)

{

cout << "\nThis program demonstrates the use of cv::CascadeClassifier class to detect objects (Face + eyes). You can use Haar or LBP features.\n"

"This classifier can recognize many kinds of rigid objects, once the appropriate classifier is trained.\n"

"It's most known use is for faces.\n"

"Usage:\n"

<< argv[0]

<< " [--cascade=<cascade\_path> this is the primary trained classifier such as frontal face]\n"

" [--nested-cascade[=nested\_cascade\_path this an optional secondary classifier such as eyes]]\n"

" [--scale=<image scale greater or equal to 1, try 1.3 for example>]\n"

" [--try-flip]\n"

" [filename|camera\_index]\n\n"

"example:\n"

<< argv[0]

<< " --cascade=\"data/haarcascades/haarcascade\_frontalface\_alt.xml\" --nested-cascade=\"data/haarcascades/haarcascade\_eye\_tree\_eyeglasses.xml\" --scale=1.3\n\n"

"During execution:\n\tHit any key to quit.\n"

"\tUsing OpenCV version " << [CV\_VERSION](https://docs.opencv.org/4.5.1/da/dd7/core_2include_2opencv2_2core_2version_8hpp.html" \l "a4f050efda432e041621b3af3df140adc) << "\n" << endl;

}

void detectAndDraw( [Mat](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html)& img, [CascadeClassifier](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html)& cascade,

[CascadeClassifier](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html)& nestedCascade,

double [scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html" \l "ae55d1c89ff5761730174745401162743), bool tryflip );

string cascadeName;

string nestedCascadeName;

int main( int argc, const char\*\* argv )

{

[VideoCapture](https://docs.opencv.org/4.5.1/d8/dfe/classcv_1_1VideoCapture.html) capture;

[Mat](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html) frame, image;

string inputName;

bool tryflip;

[CascadeClassifier](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html) cascade, nestedCascade;

double [scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743);

[cv::CommandLineParser](https://docs.opencv.org/4.5.1/d0/d2e/classcv_1_1CommandLineParser.html) parser(argc, argv,

"{help h||}"

"{cascade|data/haarcascades/haarcascade\_frontalface\_alt.xml|}"

"{nested-cascade|data/haarcascades/haarcascade\_eye\_tree\_eyeglasses.xml|}"

"{scale|1|}{try-flip||}{@filename||}"

);

if (parser.[has](https://docs.opencv.org/4.5.1/d0/d2e/classcv_1_1CommandLineParser.html" \l "a8c41cf242b2ad8c0472e3b57893ef200)("help"))

{

help(argv);

return 0;

}

cascadeName = parser.[get](https://docs.opencv.org/4.5.1/d0/d2e/classcv_1_1CommandLineParser.html" \l "a190215863579a6d244dccfd261ee0224)<string>("cascade");

nestedCascadeName = parser.[get](https://docs.opencv.org/4.5.1/d0/d2e/classcv_1_1CommandLineParser.html#a190215863579a6d244dccfd261ee0224)<string>("nested-cascade");

scale = parser.[get](https://docs.opencv.org/4.5.1/d0/d2e/classcv_1_1CommandLineParser.html#a190215863579a6d244dccfd261ee0224)<double>("scale");

if (scale < 1)

scale = 1;

tryflip = parser.[has](https://docs.opencv.org/4.5.1/d0/d2e/classcv_1_1CommandLineParser.html#a8c41cf242b2ad8c0472e3b57893ef200)("try-flip");

inputName = parser.[get](https://docs.opencv.org/4.5.1/d0/d2e/classcv_1_1CommandLineParser.html#a190215863579a6d244dccfd261ee0224)<string>("@filename");

if (!parser.[check](https://docs.opencv.org/4.5.1/d0/d2e/classcv_1_1CommandLineParser.html" \l "ad6b244a52f3b90da6fc463341c8a486f)())

{

parser.[printErrors](https://docs.opencv.org/4.5.1/d0/d2e/classcv_1_1CommandLineParser.html" \l "a70c31969ec5259b8c093d11a80c34f9f)();

return 0;

}

if (!nestedCascade.[load](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html" \l "a1a5884c8cc749422f9eb77c2471958bc)([samples::findFileOrKeep](https://docs.opencv.org/4.5.1/d6/dba/group__core__utils__samples.html" \l "ga4a37595bcd267558bd6c176c7307f050)(nestedCascadeName)))

cerr << "WARNING: Could not load classifier cascade for nested objects" << endl;

if (!cascade.[load](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html#a1a5884c8cc749422f9eb77c2471958bc)([samples::findFile](https://docs.opencv.org/4.5.1/d6/dba/group__core__utils__samples.html" \l "ga3a33b00033b46c698ff6340d95569c13)(cascadeName)))

{

cerr << "ERROR: Could not load classifier cascade" << endl;

help(argv);

return -1;

}

if( inputName.empty() || (isdigit(inputName[0]) && inputName.size() == 1) )

{

int camera = inputName.empty() ? 0 : inputName[0] - '0';

if(!capture.[open](https://docs.opencv.org/4.5.1/d8/dfe/classcv_1_1VideoCapture.html" \l "a614a1702e15f42ede5100014ce7f48ed)(camera))

{

cout << "Capture from camera #" << camera << " didn't work" << endl;

return 1;

}

}

else if (!inputName.empty())

{

image = [imread](https://docs.opencv.org/4.5.1/d4/da8/group__imgcodecs.html" \l "ga288b8b3da0892bd651fce07b3bbd3a56)([samples::findFileOrKeep](https://docs.opencv.org/4.5.1/d6/dba/group__core__utils__samples.html#ga4a37595bcd267558bd6c176c7307f050)(inputName), [IMREAD\_COLOR](https://docs.opencv.org/4.5.1/d8/d6a/group__imgcodecs__flags.html" \l "gga61d9b0126a3e57d9277ac48327799c80af660544735200cbe942eea09232eb822));

if (image.[empty](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html" \l "abbec3525a852e77998aba034813fded4)())

{

if (!capture.[open](https://docs.opencv.org/4.5.1/d8/dfe/classcv_1_1VideoCapture.html#a614a1702e15f42ede5100014ce7f48ed)([samples::findFileOrKeep](https://docs.opencv.org/4.5.1/d6/dba/group__core__utils__samples.html#ga4a37595bcd267558bd6c176c7307f050)(inputName)))

{

cout << "Could not read " << inputName << endl;

return 1;

}

}

}

else

{

image = [imread](https://docs.opencv.org/4.5.1/d4/da8/group__imgcodecs.html#ga288b8b3da0892bd651fce07b3bbd3a56)([samples::findFile](https://docs.opencv.org/4.5.1/d6/dba/group__core__utils__samples.html#ga3a33b00033b46c698ff6340d95569c13)("lena.jpg"), [IMREAD\_COLOR](https://docs.opencv.org/4.5.1/d8/d6a/group__imgcodecs__flags.html#gga61d9b0126a3e57d9277ac48327799c80af660544735200cbe942eea09232eb822));

if (image.[empty](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html#abbec3525a852e77998aba034813fded4)())

{

cout << "Couldn't read lena.jpg" << endl;

return 1;

}

}

if( capture.[isOpened](https://docs.opencv.org/4.5.1/d8/dfe/classcv_1_1VideoCapture.html" \l "a9d2ca36789e7fcfe7a7be3b328038585)() )

{

cout << "Video capturing has been started ..." << endl;

for(;;)

{

capture >> frame;

if( frame.empty() )

break;

[Mat](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html) frame1 = frame.[clone](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html" \l "adff2ea98da45eae0833e73582dd4a660)();

detectAndDraw( frame1, cascade, nestedCascade, scale, tryflip );

char c = (char)[waitKey](https://docs.opencv.org/4.5.1/d7/dfc/group__highgui.html" \l "ga5628525ad33f52eab17feebcfba38bd7)(10);

if( c == 27 || c == 'q' || c == 'Q' )

break;

}

}

else

{

cout << "Detecting face(s) in " << inputName << endl;

if( !image.[empty](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html#abbec3525a852e77998aba034813fded4)() )

{

detectAndDraw( image, cascade, nestedCascade, scale, tryflip );

[waitKey](https://docs.opencv.org/4.5.1/d7/dfc/group__highgui.html#ga5628525ad33f52eab17feebcfba38bd7)(0);

}

else if( !inputName.empty() )

{

/\* assume it is a text file containing the

list of the image filenames to be processed - one per line \*/

FILE\* f = fopen( inputName.c\_str(), "rt" );

if( f )

{

char buf[1000+1];

while( fgets( buf, 1000, f ) )

{

int len = (int)strlen(buf);

while( len > 0 && isspace(buf[len-1]) )

len--;

buf[len] = '\0';

cout << "file " << buf << endl;

image = [imread](https://docs.opencv.org/4.5.1/d4/da8/group__imgcodecs.html#ga288b8b3da0892bd651fce07b3bbd3a56)( buf, 1 );

if( !image.[empty](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html#abbec3525a852e77998aba034813fded4)() )

{

detectAndDraw( image, cascade, nestedCascade, scale, tryflip );

char c = (char)[waitKey](https://docs.opencv.org/4.5.1/d7/dfc/group__highgui.html#ga5628525ad33f52eab17feebcfba38bd7)(0);

if( c == 27 || c == 'q' || c == 'Q' )

break;

}

else

{

cerr << "Aw snap, couldn't read image " << buf << endl;

}

}

fclose(f);

}

}

}

return 0;

}

void detectAndDraw( [Mat](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html)& img, [CascadeClassifier](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html)& cascade,

[CascadeClassifier](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html)& nestedCascade,

double [scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743), bool tryflip )

{

double t = 0;

vector<Rect> faces, faces2;

const static [Scalar](https://docs.opencv.org/4.5.1/d1/da0/classcv_1_1Scalar__.html) colors[] =

{

[Scalar](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html" \l "ga599fe92e910c027be274233eccad7beb)(255,0,0),

[Scalar](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html#ga599fe92e910c027be274233eccad7beb)(255,128,0),

[Scalar](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html#ga599fe92e910c027be274233eccad7beb)(255,255,0),

[Scalar](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html#ga599fe92e910c027be274233eccad7beb)(0,255,0),

[Scalar](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html#ga599fe92e910c027be274233eccad7beb)(0,128,255),

[Scalar](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html#ga599fe92e910c027be274233eccad7beb)(0,255,255),

[Scalar](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html#ga599fe92e910c027be274233eccad7beb)(0,0,255),

[Scalar](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html#ga599fe92e910c027be274233eccad7beb)(255,0,255)

};

[Mat](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html) gray, smallImg;

[cvtColor](https://docs.opencv.org/4.5.1/d8/d01/group__imgproc__color__conversions.html" \l "ga397ae87e1288a81d2363b61574eb8cab)( img, gray, [COLOR\_BGR2GRAY](https://docs.opencv.org/4.5.1/d8/d01/group__imgproc__color__conversions.html" \l "gga4e0972be5de079fed4e3a10e24ef5ef0a353a4b8db9040165db4dacb5bcefb6ea) );

double fx = 1 / [scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743);

[resize](https://docs.opencv.org/4.5.1/d5/df1/group__imgproc__hal__functions.html" \l "ga2fe39d2201b12e1b961ca56b2aff9ff2)( gray, smallImg, [Size](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html" \l "ga346f563897249351a34549137c8532a0)(), fx, fx, [INTER\_LINEAR\_EXACT](https://docs.opencv.org/4.5.1/da/d54/group__imgproc__transform.html" \l "gga5bb5a1fea74ea38e1a5445ca803ff121ac00f4a8155563cdc23437fc0959da935) );

[equalizeHist](https://docs.opencv.org/4.5.1/d6/dc7/group__imgproc__hist.html" \l "ga7e54091f0c937d49bf84152a16f76d6e)( smallImg, smallImg );

t = (double)[getTickCount](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html" \l "gae73f58000611a1af25dd36d496bf4487)();

cascade.[detectMultiScale](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html" \l "aaf8181cb63968136476ec4204ffca498)( smallImg, faces,

1.1, 2, 0

//|CASCADE\_FIND\_BIGGEST\_OBJECT

//|CASCADE\_DO\_ROUGH\_SEARCH

|[CASCADE\_SCALE\_IMAGE](https://docs.opencv.org/4.5.1/d5/d54/group__objdetect.html" \l "gga3f9a572bf02ebe19f6652ce7cadadc7ca1d35c5234686c05c075d224efc91d673),

[Size](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html#ga346f563897249351a34549137c8532a0)(30, 30) );

if( tryflip )

{

[flip](https://docs.opencv.org/4.5.1/d2/de8/group__core__array.html" \l "gaca7be533e3dac7feb70fc60635adf441)(smallImg, smallImg, 1);

cascade.[detectMultiScale](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html#aaf8181cb63968136476ec4204ffca498)( smallImg, faces2,

1.1, 2, 0

//|CASCADE\_FIND\_BIGGEST\_OBJECT

//|CASCADE\_DO\_ROUGH\_SEARCH

|[CASCADE\_SCALE\_IMAGE](https://docs.opencv.org/4.5.1/d5/d54/group__objdetect.html#gga3f9a572bf02ebe19f6652ce7cadadc7ca1d35c5234686c05c075d224efc91d673),

[Size](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html#ga346f563897249351a34549137c8532a0)(30, 30) );

for( vector<Rect>::const\_iterator r = faces2.begin(); r != faces2.end(); ++r )

{

faces.push\_back([Rect](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html" \l "ga11d95de507098e90bad732b9345402e8)(smallImg.[cols](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html" \l "aa3e5a47585c9ef6a0842556739155e3e) - r->x - r->width, r->y, r->width, r->height));

}

}

t = (double)[getTickCount](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html#gae73f58000611a1af25dd36d496bf4487)() - t;

printf( "detection time = %g ms\n", t\*1000/[getTickFrequency](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html" \l "ga705441a9ef01f47acdc55d87fbe5090c)());

for ( size\_t i = 0; i < faces.size(); i++ )

{

[Rect](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html) r = faces[i];

[Mat](https://docs.opencv.org/4.5.1/d3/d63/classcv_1_1Mat.html) smallImgROI;

vector<Rect> nestedObjects;

[Point](https://docs.opencv.org/4.5.1/db/d4e/classcv_1_1Point__.html) center;

[Scalar](https://docs.opencv.org/4.5.1/d1/da0/classcv_1_1Scalar__.html) color = colors[i%8];

int radius;

double aspect\_ratio = (double)r.[width](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html" \l "a6c16a3bce912faa4fe5be42d7f1b53fe)/r.[height](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html" \l "a6fed06513cedd76652389e38c7b1222e);

if( 0.75 < aspect\_ratio && aspect\_ratio < 1.3 )

{

center.[x](https://docs.opencv.org/4.5.1/db/d4e/classcv_1_1Point__.html" \l "a4c96fa7bdbfe390be5ed356edb274ff3) = [cvRound](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html" \l "ga085eca238176984a0b72df2818598d85)((r.[x](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html" \l "a2cadfdc3b4b7dbf8085622b27e044572) + r.[width](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6c16a3bce912faa4fe5be42d7f1b53fe)\*0.5)\*[scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743));

center.[y](https://docs.opencv.org/4.5.1/db/d4e/classcv_1_1Point__.html" \l "a157337197338ff199e5df1a393022f15) = [cvRound](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html#ga085eca238176984a0b72df2818598d85)((r.[y](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html" \l "a6a4860e984df1752623b6ce2a8bde73a) + r.[height](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6fed06513cedd76652389e38c7b1222e)\*0.5)\*[scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743));

radius = [cvRound](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html#ga085eca238176984a0b72df2818598d85)((r.[width](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6c16a3bce912faa4fe5be42d7f1b53fe) + r.[height](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6fed06513cedd76652389e38c7b1222e))\*0.25\*[scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743));

[circle](https://docs.opencv.org/4.5.1/d9/db7/group__datasets__gr.html" \l "gga610754124ced68d1f05760b5948fbb76a6f0d8b2d9e3e947b2a5c1eff9e81ee95)( img, center, radius, color, 3, 8, 0 );

}

else

[rectangle](https://docs.opencv.org/4.5.1/d6/d6e/group__imgproc__draw.html" \l "ga07d2f74cadcf8e305e810ce8eed13bc9)( img, [Point](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html" \l "ga1e83eafb2d26b3c93f09e8338bcab192)([cvRound](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html#ga085eca238176984a0b72df2818598d85)(r.[x](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a2cadfdc3b4b7dbf8085622b27e044572)\*[scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743)), [cvRound](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html#ga085eca238176984a0b72df2818598d85)(r.[y](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6a4860e984df1752623b6ce2a8bde73a)\*[scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743))),

[Point](https://docs.opencv.org/4.5.1/db/d4e/classcv_1_1Point__.html)([cvRound](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html#ga085eca238176984a0b72df2818598d85)((r.[x](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a2cadfdc3b4b7dbf8085622b27e044572) + r.[width](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6c16a3bce912faa4fe5be42d7f1b53fe)-1)\*[scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743)), [cvRound](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html#ga085eca238176984a0b72df2818598d85)((r.[y](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6a4860e984df1752623b6ce2a8bde73a) + r.[height](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6fed06513cedd76652389e38c7b1222e)-1)\*[scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743))),

color, 3, 8, 0);

if( nestedCascade.[empty](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html" \l "a1753ebe58554fe0673ce46cb4e83f08a)() )

continue;

smallImgROI = smallImg( r );

nestedCascade.[detectMultiScale](https://docs.opencv.org/4.5.1/d1/de5/classcv_1_1CascadeClassifier.html#aaf8181cb63968136476ec4204ffca498)( smallImgROI, nestedObjects,

1.1, 2, 0

//|CASCADE\_FIND\_BIGGEST\_OBJECT

//|CASCADE\_DO\_ROUGH\_SEARCH

//|CASCADE\_DO\_CANNY\_PRUNING

|[CASCADE\_SCALE\_IMAGE](https://docs.opencv.org/4.5.1/d5/d54/group__objdetect.html#gga3f9a572bf02ebe19f6652ce7cadadc7ca1d35c5234686c05c075d224efc91d673),

[Size](https://docs.opencv.org/4.5.1/dc/d84/group__core__basic.html#ga346f563897249351a34549137c8532a0)(30, 30) );

for ( size\_t j = 0; j < nestedObjects.size(); j++ )

{

[Rect](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html) nr = nestedObjects[j];

center.[x](https://docs.opencv.org/4.5.1/db/d4e/classcv_1_1Point__.html#a4c96fa7bdbfe390be5ed356edb274ff3) = [cvRound](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html#ga085eca238176984a0b72df2818598d85)((r.[x](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a2cadfdc3b4b7dbf8085622b27e044572) + nr.[x](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a2cadfdc3b4b7dbf8085622b27e044572) + nr.[width](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6c16a3bce912faa4fe5be42d7f1b53fe)\*0.5)\*[scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743));

center.[y](https://docs.opencv.org/4.5.1/db/d4e/classcv_1_1Point__.html#a157337197338ff199e5df1a393022f15) = [cvRound](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html#ga085eca238176984a0b72df2818598d85)((r.[y](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6a4860e984df1752623b6ce2a8bde73a) + nr.[y](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6a4860e984df1752623b6ce2a8bde73a) + nr.[height](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6fed06513cedd76652389e38c7b1222e)\*0.5)\*[scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743));

radius = [cvRound](https://docs.opencv.org/4.5.1/db/de0/group__core__utils.html#ga085eca238176984a0b72df2818598d85)((nr.[width](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6c16a3bce912faa4fe5be42d7f1b53fe) + nr.[height](https://docs.opencv.org/4.5.1/d2/d44/classcv_1_1Rect__.html#a6fed06513cedd76652389e38c7b1222e))\*0.25\*[scale](https://docs.opencv.org/4.5.1/d6/d84/namespacecv_1_1quality_1_1quality__utils.html#ae55d1c89ff5761730174745401162743));

[circle](https://docs.opencv.org/4.5.1/d9/db7/group__datasets__gr.html#gga610754124ced68d1f05760b5948fbb76a6f0d8b2d9e3e947b2a5c1eff9e81ee95)( img, center, radius, color, 3, 8, 0 );

}

}

[imshow](https://docs.opencv.org/4.5.1/d7/dfc/group__highgui.html" \l "ga453d42fe4cb60e5723281a89973ee563)( "result", img );

Приложение Б

**Графический материал**

Внешний вид графического материала, выполненного на отдельных листах представлен на рисунках Б.1 – Б.5.

Перечень графического материала:

лист 1: Постановка задачи (рис Б.1),

лист 2: Концептуальная модель предметной области (рис. Б.2),

лист 3: Диаграмма состояний пользовательского интерфейса (рис. Б.3),

лист 4: Диаграмма последовательности, описывающая поток, осуществляющий распознавание эмоций (рис. Б.4),

лист 5: Диаграмма классов (рис. Б.5).



Рисунок Б1 – Постановка задачи



Рисунок Б.2 – Концептуальная модель предметной области



Рисунок Б.3 - Диаграмма состояний пользовательского интерфейса



Рисунок Б.4 – Диаграмма последовательности, описывающая поток, осуществляющий распознавание эмоций



Рисунок Б.5 – Диаграмма классов