**МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ**

**Государственное бюджетное профессиональное образовательное учреждение Московской области «Люберецкий техникум имени Героя Советского Союза, летчика-космонавта Ю.А.Гагарина»**

**ОТЧЕТ**

**УЧЕБНОЙ ПРАКТИКИ**

                                Будаев Владимир Константинович

(Фамилия, имя, отчество студента)

по профессиональному модулю

         *ПМ.01 «Разработка модулей программного обеспечения*

                   *для компьютерных систем»*

Специальность \_*09.02.07 "Информационные системы и программирование"*

Код, название

Курс \_3\_ Группа № \_195\_

Период практики с «12» ноября 2021 г. по «24» ноября 2021 г.

Руководитель практики

от техникума                                     /Ревнивцев М.В.  /

                                    /Жирнова Ю.В.    /

Люберцы 202

Содержание

Введение                                                                                                                                                     3

Изучение предметной области                                                                                                           5

Изучение программных средств                                                                                                        6

OpenCV                                                                                                                                                   8

Discord Application Modules                                                                                                                  18

Приложение WPF                                                                                                                                20

Выводы                                                                                                                                         32

[Список](https://docs.google.com/document/d/1WxC7U7BJg-haKaleu3ikniprsEiU8tyy/edit#heading=h.3dy6vkm) литературы                                                                                                                          33

Дневник учебной практики                                                                                                               34

# Введение

Учебная практика представляет собой важнейшую составную часть      учебного процесса по подготовке специалистов, способствует повышению общего уровня профессиональной подготовки, закреплению и углублению полученных теоретических знаний по дисциплинам.

Целью учебной практики является реализация компьютерного зрения с помощью библиотеки OpenCV и разработка мини-приложений WPF на языке программирования C#.

Для достижения данной цели требуется решить следующие задачи:

1. Знакомство с литературой
2. Изучить библиотеку OpenCV
3. Реализация приложения на WPF
4. Пройти курс «Discord Application Modules»

В ходе учебной практики планируется освоение следующего вида деятельности: «Разработка модулей программного обеспечения для компьютерных систем» (ПК):

ПК 1.1 Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.

ПК 1.2 Разрабатывать программные модули в соответствии с техническим заданием.

ПК 1.3 Выполнять отладку программных модулей с использованием специализированных программных средств

ПК 1.4 Выполнять тестирование программных модулей

ПК 1.5 Осуществлять рефакторинг и оптимизацию программного кода

ПК 1.6 Разрабатывать модули программного обеспечения для мобильных платформ.

ПК 2.1. Разрабатывать требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент.

ПК 2.2. Выполнять интеграцию модулей в программное обеспечение.

ПК 2.3. Выполнять отладку программного модуля с использованием специализированных программных средств.

ПК 2.4. Осуществлять разработку тестовых наборов и тестовых сценариев для программного обеспечения.

ПК 2.5. Производить инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования.

# Изучение предметной области

**Предметная область** - [множество](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE) всех предметов, [свойства](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%BE) которых и отношения между которыми рассматриваются в [научной теории](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D1%83%D1%87%D0%BD%D0%B0%D1%8F_%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F).

**OpenCV** — это open source библиотека компьютерного зрения, которая предназначена для анализа, классификации и обработки изображений. Широко используется в таких языках как C, C++, Python и Java.

* Для утверждения общего стандартного интерфейса компьютерного зрения для приложений в этой области. Для способствования росту числа таких приложений и создания новых моделей использования PC.
* Сделать платформы Intel привлекательными для разработчиков таких приложений за счёт дополнительного ускорения OpenCV с помощью Intel® Performance Libraries (Сейчас включают IPP (низкоуровневые библиотеки для обработки сигналов, изображений, а также медиа-кодеки) и MKL (специальная версия LAPACK и FFTPack)). OpenCV способна автоматически обнаруживать присутствие IPP и MKL и использовать их для ускорения обработки.

# 

# 

# Изучение программных средств

# Microsoft SQL Server Management Studio 18 - Утилита из Microsoft SQL Server 2005 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server. Утилита включает скриптовый редактор и графическую программу, которая работает с объектами и настройками сервера.

Достоинства:

* **Полная Web ориентированность.** Осуществление запросов, анализ и управление через Web. Использование языка XML для обмена информацией между удаленными системами.
* **Масштабируемость и надежность.** SQL Server 2018 обеспечивает практически неограниченный рост объемов хранения за счет увеличения надежности и масштабируемости системы, используя все преимущества мультипроцессорной обработки файлов. Это безопасная, надежная, масштабируемая платформа, защищающая информацию в приложениях и повышающая её доступность.
* **Скорость создания решений.** SQL Server 2018 в сочетании с .NET Framework уменьшает время разработки, внедрения и выхода на рынок современных приложений, ускоряет процесс поиска информации, упрощает управление, позволяет использовать создаваемые пользователем функции в других приложениях, предоставляет широкие возможности для создания Web-приложений.
* **Возможность обработки вычислений в оперативной памяти**

(in-memory OLTP)

* **Возможность взаимодействия с публичным облаком Windows Azure**. Облачное резервирование локальных систем и аварийное восстановление из облака повышает отказоустойчивость критичных для бизнеса приложений.

**Microsoft Visual Studio** — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментов

Достоинства:

* **Встроенный Web-сервер**. Для обслуживания Web-приложения ASP.NET необходим Web-сервер, который будет ожидать Web-запросы и обрабатывать соответствующие страницы. Наличие в Visual Studio интегрированного Web-сервера позволяет запускать Web-сайт прямо из среды проектирования, а также повышает безопасность, исключая вероятность получения доступа к тестовому Web-сайту с какого-нибудь внешнего компьютера, поскольку тестовый сервер может принимать соединения только с локального компьютера.
* **Поддержка множества языков при разработке**. Visual Studio позволяет писать код на своем языке или любых других предпочитаемых языках, используя все время один и тот же интерфейс (IDE). Более того, Visual Studio также еще позволяет создавать Web-страницы на разных языках, но помещать их все в одно и то же Web-приложение.
* **Интуитивный стиль кодирования**. По умолчанию Visual Studio форматирует код по мере его ввода, автоматически вставляя необходимые отступы и применяя цветовое кодирование для выделения элементов типа комментариев. Такие незначительные отличия делают код более удобным для чтения и менее подверженным ошибкам.
* **Более высокая скорость разработки**. Многие из функциональных возможностей Visual Studio направлены на то, чтобы помогать разработчику делать свою работу как можно быстрее. Удобные функции, вроде функции IntelliSense (которая умеет перехватывать ошибки и предлагать правильные варианты), функции поиска и замены (которая позволяет отыскивать ключевые слова как в одном файле, так и во всем проекте) и функции автоматического добавления и удаления комментариев (которая может временно скрывать блоки кода), позволяют разработчику работать быстро и эффективно.

**PyCharm** — интегрированная среда разработки для языка программирования Python. Предоставляет средства для анализа кода, графический отладчик, инструмент для запуска юнит-тестов и поддерживает веб-разработку на Django. PyCharm разработана компанией JetBrains на основе IntelliJ IDEA.

Достоинства:

* **Простая организация проектов.** Очень просто создавать проекты и открывать уже существующие: «пайчарм» буквально в два клика позволяет приступить к редактированию кода. Не нужно импортировать проект в workspace или делать что-то подобное.
* **Удобный автокомпилит.** Автокомплит работает мгновенно, не нужно вызывать его «хоткеем». Работает не только в привычных местах, но и в шаблонах Джанго, для подстановки путей к файлам, к тому же очень шустрый.

**Приятный интерфейс.** PyCharm вобрал в себя кучу приятных фич: двустрочные вкладки, вертикальные направляющие для выравнивания текста, быстрый и «неглючный кодфолдинг», логичное расположение элементов интерфейса, который не выглядит перегруженным.

## OpenCV

1. Теория

Файл Contour\_main.cpp - это код по нахождению контуров

Файл momentmain.cpp - это дополнительно использование ползунка createTrackbar

Файл VideoTH.cpp - это оператор Собеля

Оператор Собеля(рис.1) — дискретный дифференциальный оператор, вычисляющий приближённое значение градиента яркости изображения. Результатом применения оператора Собеля в каждой точке изображения является либо вектор градиента яркости в этой точке, либо его норма. Используется в области обработки изображений, в частности, часто применяется в алгоритмах выделения границ.

VideoCapture cap – оператор для использования веб камеры

detectMultiScale — общая функция для распознавания как лиц, так и объектов. Чтобы функция искала именно лица, мы передаём ей соответствующий каскад.

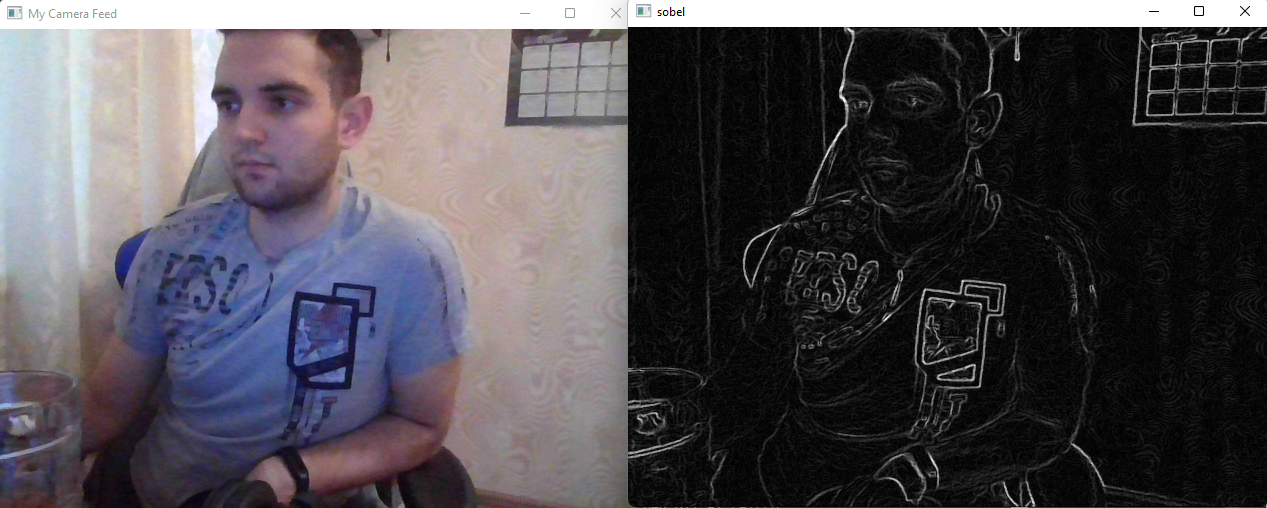


Рисунок 1.Оператор Собеля

Функция detectMultiScale принимает 4 параметра:

* + - 1. Обрабатываемое изображение в градации серого.
* Параметр scaleFactor. Некоторые лица могут быть больше других, поскольку находятся ближе, чем остальные. Этот параметр компенсирует перспективу.
* Алгоритм распознавания использует скользящее окно во время распознавания объектов. Параметр minNeighbors определяет количество объектов вокруг лица. То есть чем больше значение этого параметра, тем больше аналогичных объектов необходимо алгоритму, чтобы он определил текущий объект, как лицо. Слишком маленькое значение увеличит количество ложных срабатываний, а слишком большое сделает алгоритм более требовательным.
* minSize — непосредственно размер этих областей.

Сохранение изображения

import cv2

image = cv2.imread("./импорт/путь.расширение")

cv2.imwrite("./экспорт/путь.расширение", image)

Рисование прямоугольников

import cv2

output = image.copy()

cv2.rectangle(output, (2600, 800), (4100, 2400), (0, 255, 255), 10)

viewImage(output, "Обводим прямоугольником лицо пёсика")

Эта функция принимает 5 параметров:

* Само изображение.
* Координата верхнего левого угла (x1, y1).
* Координата нижнего правого угла (x2, y2).
* Цвет прямоугольника (GBR/RGB в зависимости от выбранной цветовой модели).
* Толщина линии прямоугольника.

Распознавание лиц (Рис. 2)

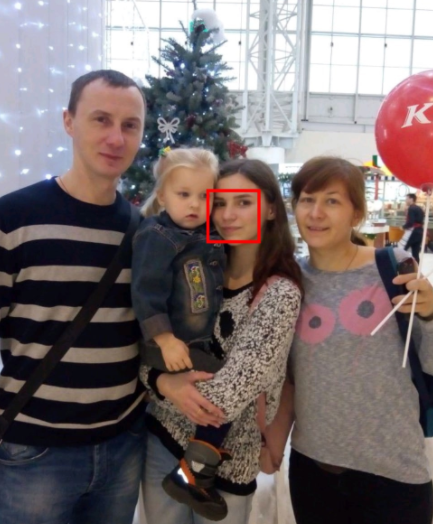


Рисунок 2. Распознавание лиц с помощью библиотеки OpenCV

Конторы и центры масс (Рис. 3)

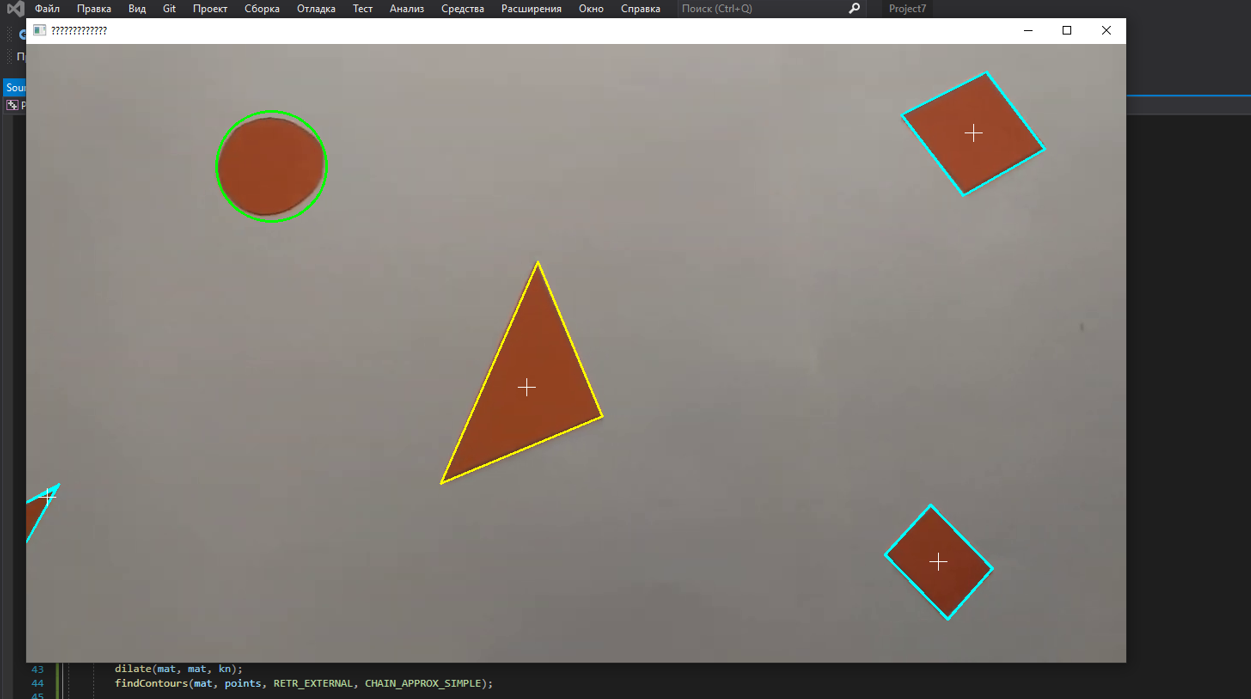


Рисунок 3. Конторы и центры масс

Распознавание дорожной разметки (Рис. 4)

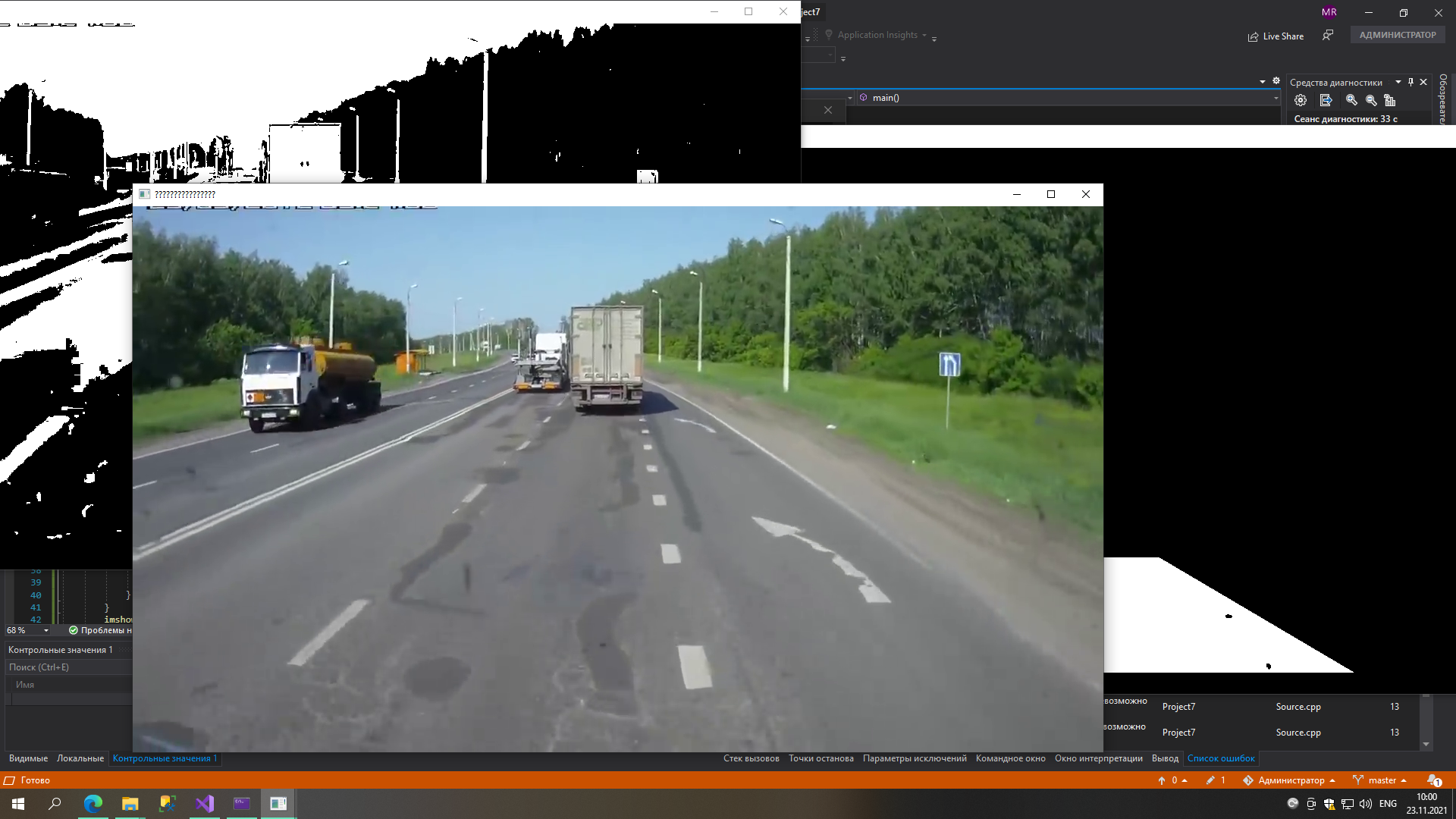


Рисунок 4. Распознавание дорожной разметки

Просмотр видео с веб камеры оператор собеля (Рис. 1)

# Оператор Canny().

Фильтр Состоит из этапов:

* Преобразование в оттенки серого - rgb2gray
* Сглаживание blur()
* Гаусс: отсекает Передали градиент - направление. рисует вместо каждого пикселя стрелочку
* Они считаются контуром, если имеют одно направление и максимальную яркость - подавление
* vector -  тип данных который позволяет хранить координаты (x и y начало контура|x и y конца контура) Есть вектор момента. Его использование позволяет определить нулевой(m00) момент (кол-во точек в контуре). Первый момент(m10)**Рефакторинг** - улучшение читаемости кода путём его переписывания не меняя функциональность. Функция - отдельный блок кода. Позволяет использовать ранее написанный код. Как найти функцию: Имя(её аргументы). ЧТобы написать функцию указываем тип данных, имя функции(тип данных аргумента имя аргумента) {тело функции}.

**Оператор Собеля** — это дискретный дифференциальный оператор, вычисляющий приближение градиента яркости изображения.

Оператор вычисляет градиент яркости изображения в каждой точке. Так находится направление наибольшего увеличения яркости и величина её изменения в этом направлении. Результат показывает, насколько «резко» или «плавно» меняется яркость изображения в каждой точке, а значит, вероятность нахождения точки на грани, а также ориентацию границы.

Класс **Mat** библиотеки OpenCV используется для хранения значений изображения. Он представляет собой n-мерный массив и используется для хранения данных изображения в градациях серого или цветных изображений, объемов вокселей, векторных полей, облаков точек

Для начала работы с библиотекой OpenCV нам необходимо создать проект в IDE Visual Studio 2022 (рисунок 1).

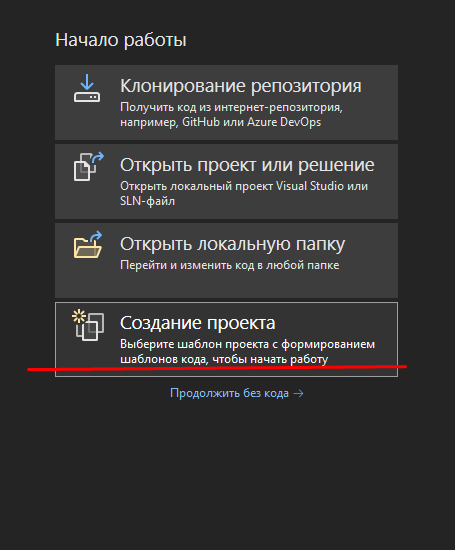


Рисунок 1. Создание проекта

Далее, на выбор будет предложено выбрать что мы хотим создать. Выбираем «Консольное приложение» и нажимаем далее (Рисунок 2).

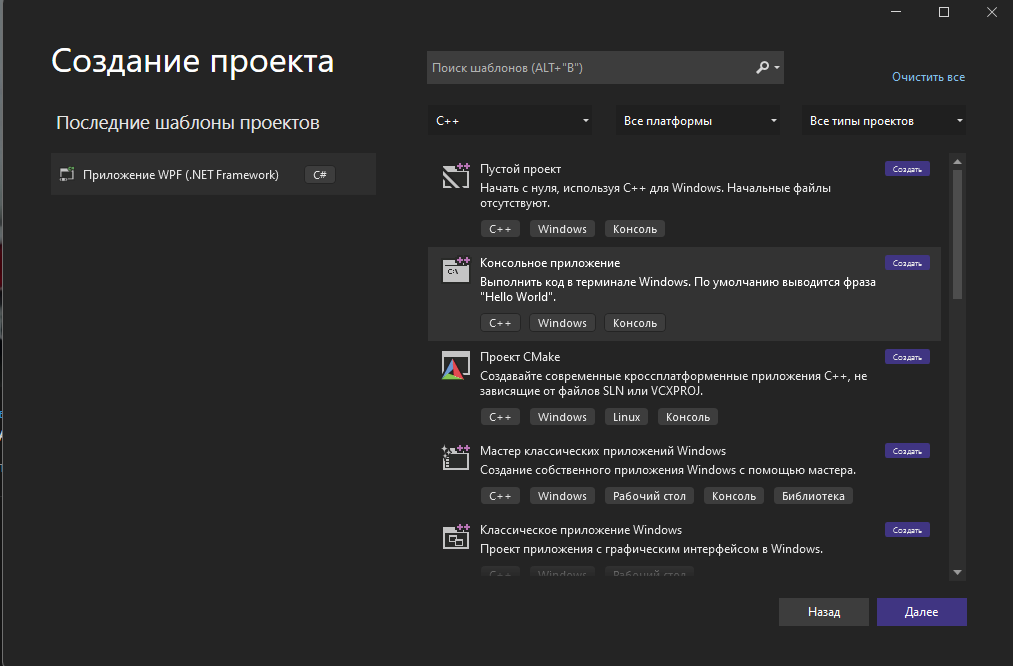


Рисунок 2. Выбор проекта

Вводим название проекта, его расположение и нажимаем кнопку «Создать» (рисунок 3).

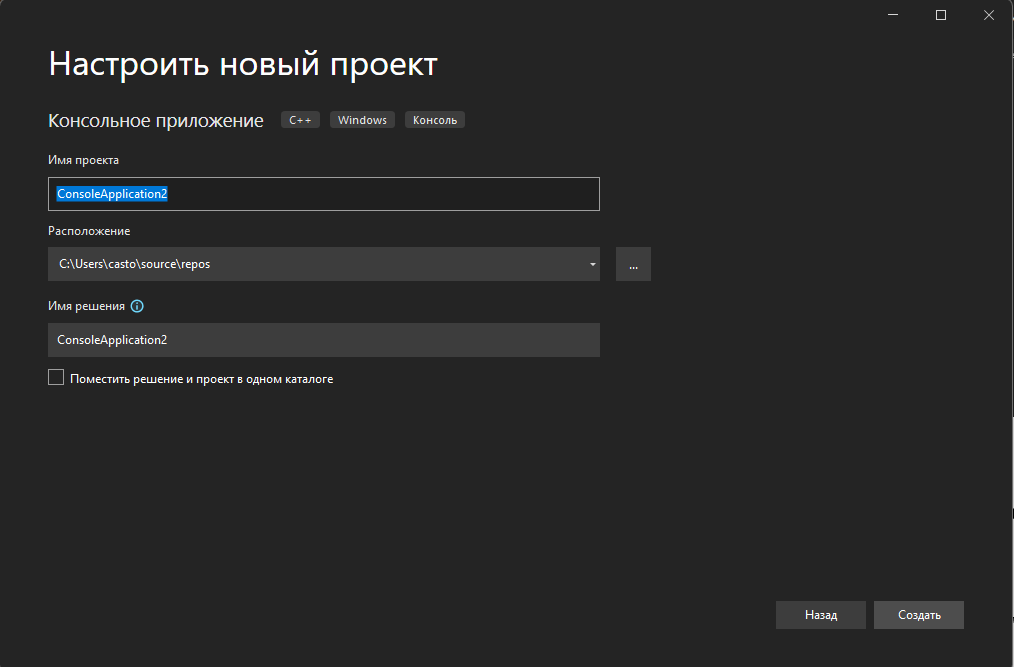


Рисунок 3. Настройка проекта

После создания проекта нам открывается возможность писать код. Весь наш код будет находиться в файле «ConsoleApplication2.cpp» (Рисунок 4).

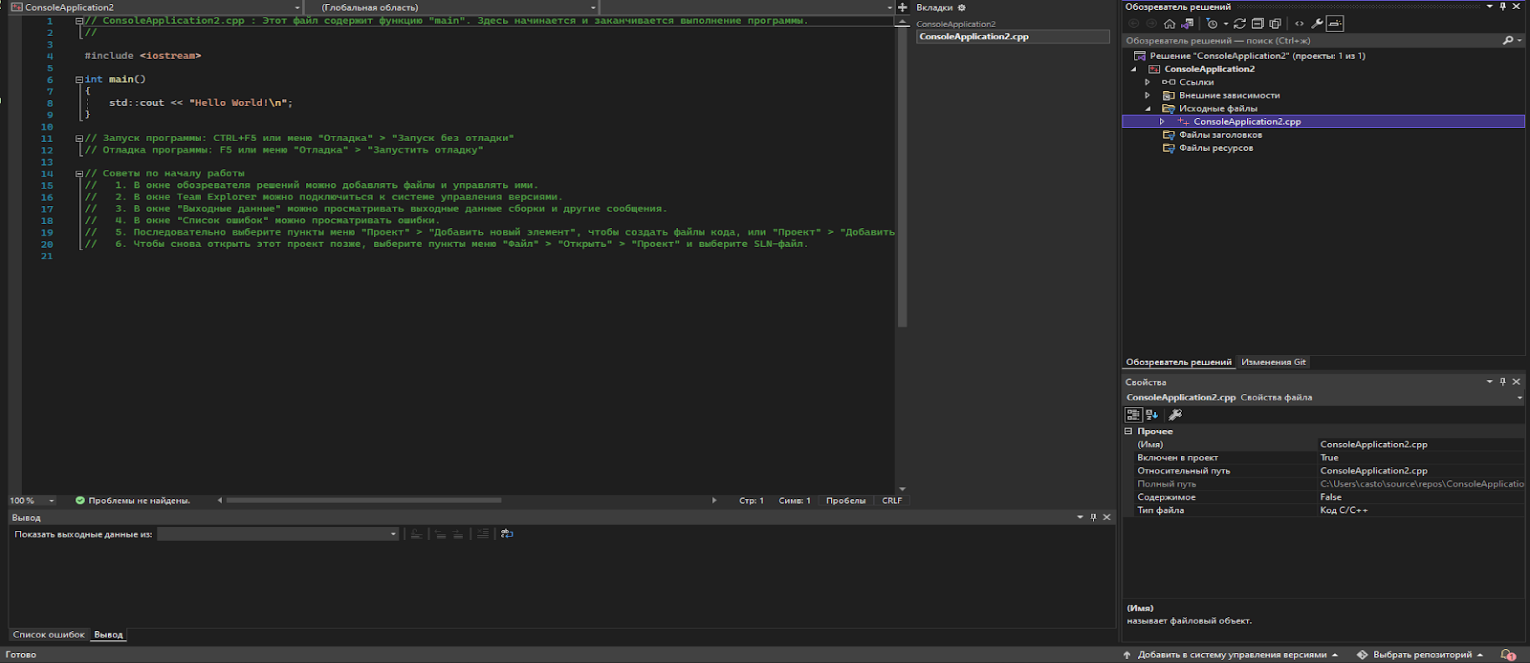


Рисунок 4. Окно с кодом

После знакомства со средой разработки, нам необходимо подключить библиотеку OpenCV. Начать нужно со скачивания этой библиотеку. Поскольку OpenCV – библиотека с открытым исходным кодом, скачать её можно [здесь](https://opencv.org/releases/) (рисунок 5).



Рисунок 5. Сайт для скачивания библиотеки

После скачивания открываем exe файл и устанавливаем в удобное место(рисунок 6).

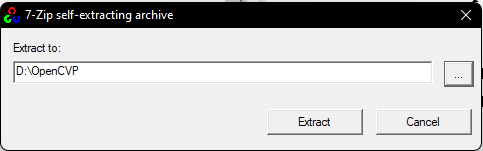


Рисунок 6. Установка библиотеки

После установки, переходим в наш проект, нажимаем правой кнопкой по  файлу «ConsoleApplication2.cpp» и выбираем кнопку «Свойства» (Рисунок 7)

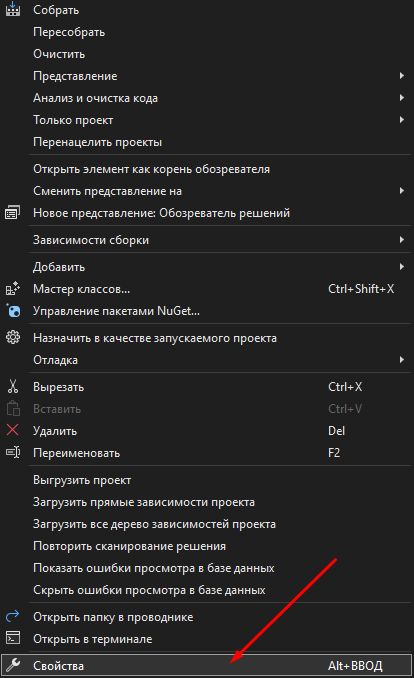


Рисунок 7. Свойства «ConsoleApplication2»

Далее переходим во вкладку C/C++ 🡪 Общие 🡪 Дополнительные каталоги включаемых файлов и вводим путь к папке «include» из библиотеки OpenCV (Рисунок 8)

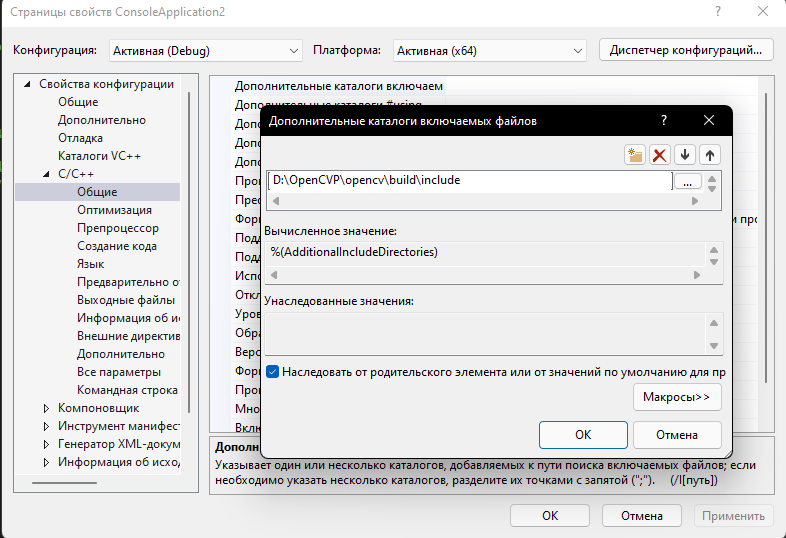


Рисунок 8. Дополнительные каталоги включаемых файлов

После этого, переходим во вкладку «Компоновщик» 🡪 Общие 🡪 Дополнительные каталоги библиотек и вводим путь к lib файлам из библиотеки OpenCV (Рисунок 9)

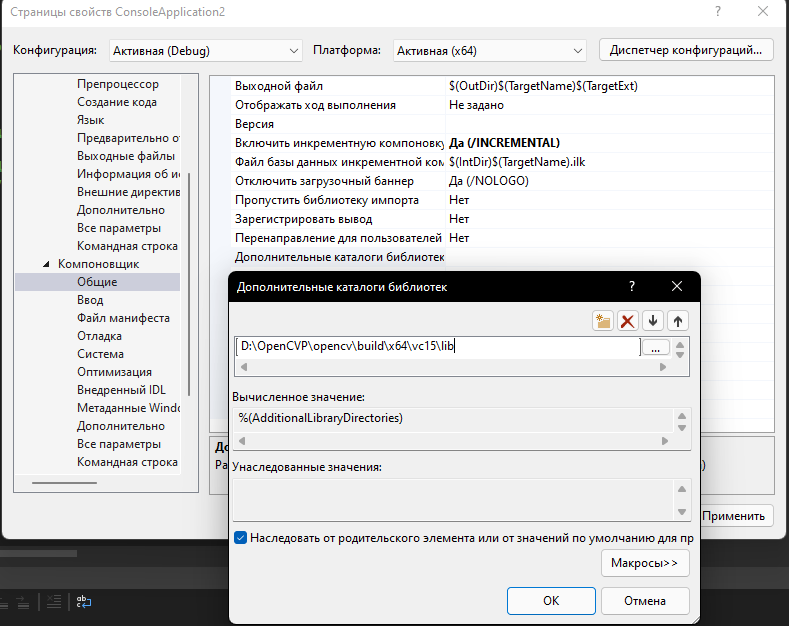


Рисунок 9. Дополнительные каталоги библиотек

Затем переходим во вкладку Ввод и в разделе «Дополнительные зависимости» прописываем названия lib файлов (Рисунок 10)

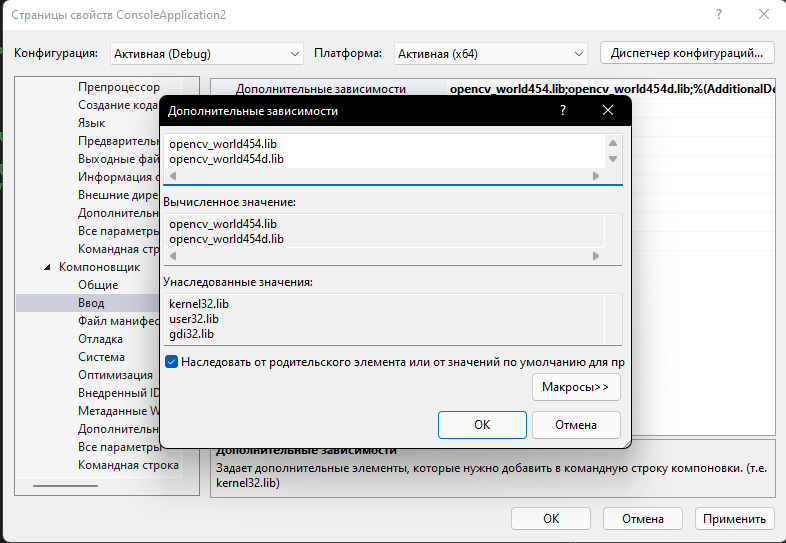


Рисунок 10. Дополнительные зависимости

# Ваш курс «Discord Application Modules»

Устанавливаем саму библиотеку.

pip install discord

Создаём файл config.py (так удобнее), и создаём там словарь.

settings = {

'token': 'Ваш токен',

'bot': 'Имя бота',

'id': Client ID бота, без кавычек,

'prefix': 'Префикс бота'

}

Создаём main-файл, название может быть любое.  
Импортируем библиотеки и наш файл конфига:

**import** discord

**from** discord.ext **import** commands

**from** config **import** settings

Создаём "тело" бота, название может быть любое:

bot = commands.Bot(command\_prefix = settings['prefix']) # Так как мы указали префикс в settings, обращаемся к словарю с ключом prefix.

Начинаем писать основной код.

@bot.command() # Не передаём аргумент pass\_context, так как он был нужен в старых версиях.

**async** **def** **hello**(ctx): # Создаём функцию и передаём аргумент ctx.

author = ctx.message.author # Объявляем переменную author и записываем туда информацию об авторе.

**await** ctx.send(f'Hello, {author.mention}!') # Выводим сообщение с упоминанием автора, обращаясь к переменной author.

В конце запускаем бота с помощью:

bot.run(settings['token']) # Обращаемся к словарю settings с ключом token, для получения токена

Сделаем вывод случайных картинок с лисами  
Для этого импортируем еще пару библиотек:

**import** json

**import** requests

Приступим к написанию команды.

@bot.command()

**async** **def** **fox**(ctx):

response = requests.get('https://some-random-api.ml/img/fox') # Get-запрос

json\_data = json.loads(response.text) # Извлекаем JSON

embed = discord.Embed(color = 0xff9900, title = 'Random Fox') # Создание Embed'a

embed.set\_image(url = json\_data['link']) # Устанавливаем картинку Embed'a

**await** ctx.send(embed = embed) # Отправляем Embed

## WPF

**Windows Presentation Foundation** — аналог WinForms, система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая подсистема в составе .NET Framework, использующая язык XAML.

Для начала работы нам потребуется IDE Visual Studio 2022. Переходим на официальный сайт Microsoft и скачиваем [Visual Studio 2022 Community Edition](https://visualstudio.microsoft.com/ru/vs/community/), она бесплатна (Рис. 11)

Рисунок 11. Скачиваем Visual Studio 2022 Community Edition

Затем запускаем exe файл, выбираем удобное место для установки и выбираем нужную среду разработки



Рисунок 12. Установка завершена

После установки SQL сервера, необходимо установить средство взаимодействия с базой данных SQL – Sql Server Management Studio (SSMS), скачивается данное приложение аналогично серверу, с официального сайта Microsoft (Рис 13.). На рисунке 4 показан процесс установки SSMS.

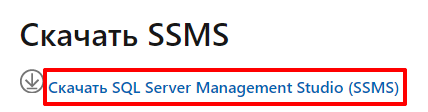


Рисунок 13. Скачивание SSMS с официального сайта Microsoft

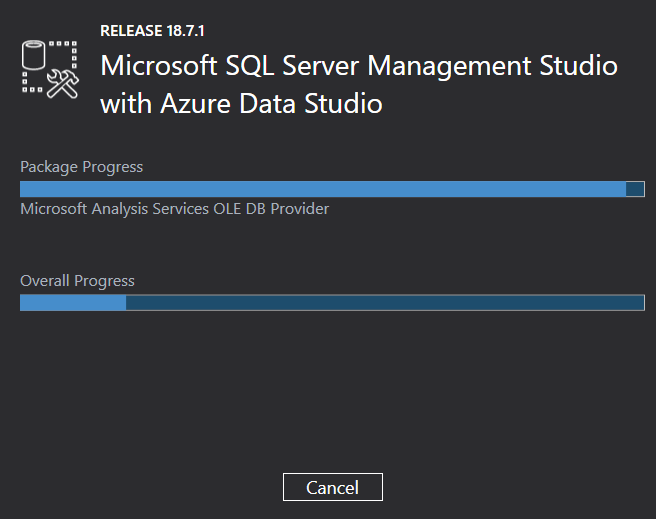


Рисунок 14. Процесс установки SSMS

При прохождении учебной практики было создано два приложения на WPF, первое это калькулятор (Рис. 15), второе личный кабинет пользователя (Рис. 16).

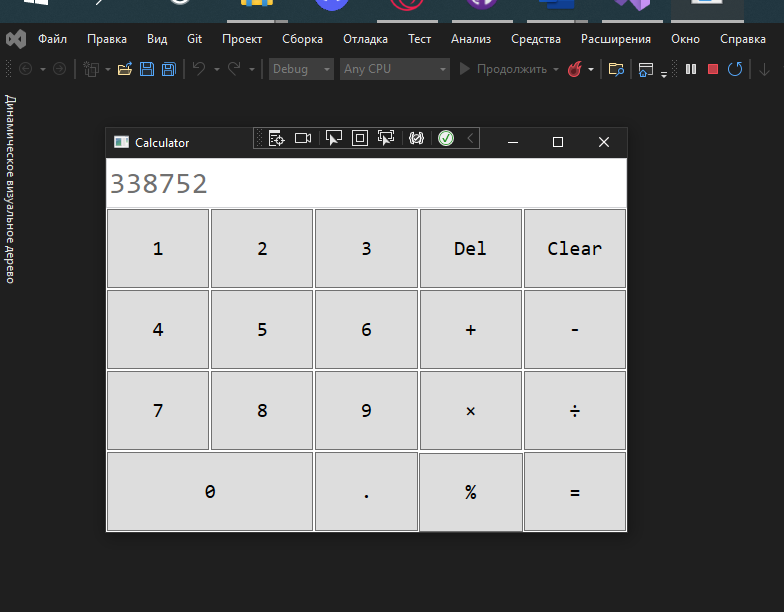


Рисунок 15. Калькулятор



Рисунок 16. Личный кабинет пользователя

При разработке калькулятора, был создан новый проект в Visual Studio на основе WPF с платформой Net.Framework.

Даём название нашему проекту и переходим к окну с кодом. В окне MainWindow.xaml с помощью кода прописываем дизайн нашего калькулятора

Затем, в файле MainWindow.xaml.cs пишем функции для нашего калькулятора. Прописываем к каждой кнопке и полю (Рис. 17)



Рисунок 17. Прописываем код для каждой кнопки калькулятора

После этого создаём новый проект, добавляем библиотеку MaterialDesignThemes c помощью «Управления пакетами NuGet»

Затем, необходимо дописать код для подключения библиотеки к проекту, для этого переходим на [официальный сайт библиотеки](http://materialdesigninxaml.net/) и копируем следующий код. Вставляем его в файл App.xaml (Рис. 18)

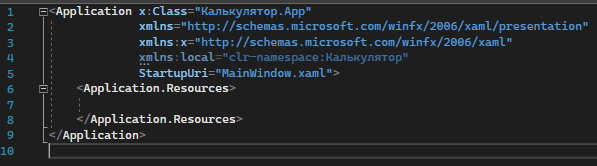


Рисунок 18. Выполненный файл App.xaml

Далее, создадим дизайн программы с помощью кода, писать его нужно в файле MainWindow.xaml(Рис. 19)

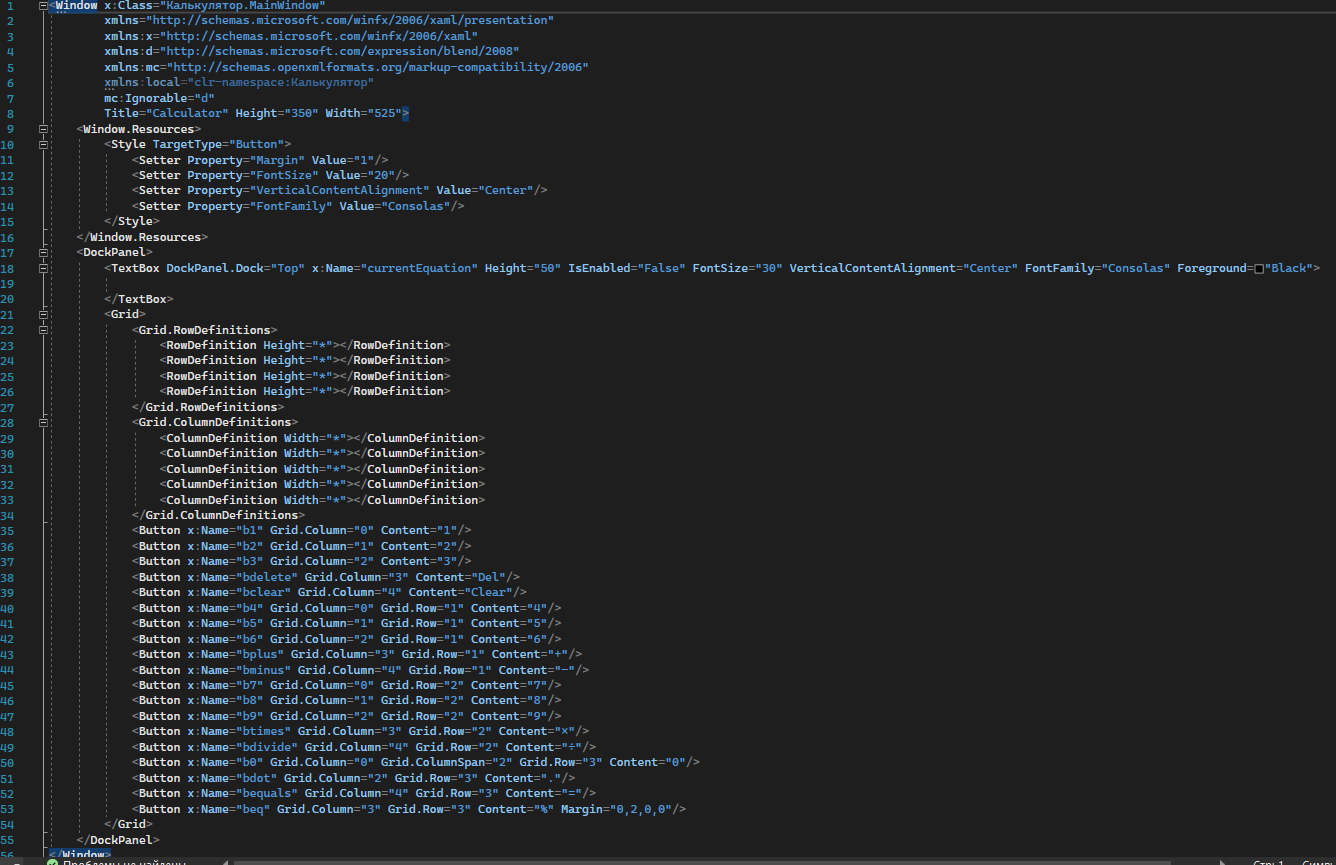


Рисунок 19. Код для дизайна калькулятора

После создания дизайна, переходим к получению данных от пользователя. Нам необходимо получить данные от пользователя и проверить их. Для этого пишем проверки вводимых данных

Далее, переходим к тестированию нашей программы. Вводим некорректные данные и видим, что наши поля для ввода «горят красным цветом». Это значит, что наши проверки работают и можно переходить к следующему этапу разработки.

Следующим этапом разработки является создание и подключение Базы данных к нашей программе, для этого потребуется скачать и установить Microsoft SQL Server Management Studio 18. Скачать эту программу можно на [официальном сайте Microsoft](https://docs.microsoft.com/ru-ru/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15#download-ssms). Показано выше.

Следуя инструкциям по установке, производим установки программы и запускаем её. Перед нами открывается окно программы. Вводим требуемые данные и переходим к созданию нашей базы данных (Рис. 20)

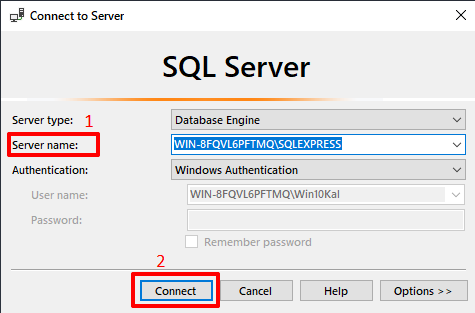


Рисунок 20. Начало работы SSMS

Затем, в обозревателе объектов выбираем «Базы данных» и создаём базу данных

После создания базы данных создаём таблицу «bd»

После создания базы данных и таблицы, переходим к подключению базы к нашему приложению. Для этого нужно нажать правой кнопкой мыши по файлу нашего проекта и нажать кнопку «Добавить» 🡪 «Создать элемент»

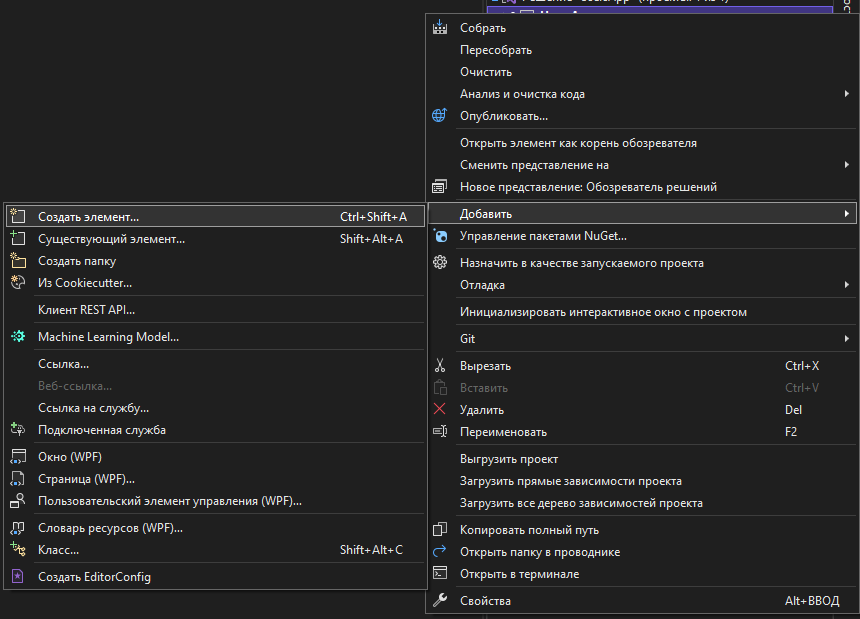


Рисунок 21. Создаём элемент

В окне «Добавление нового элемента» в поле «Поиск» вводим «edm» и добавляем Модель ADO.NET.EDM (Рис. 22)

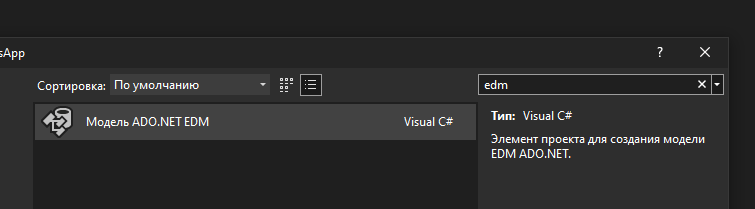


Рисунок 22. Добавляем Модель ADO.NET.EDM

Далее, следуя инструкция, вводим имя нашего сервера, выбираем нашу БД, выбираем таблицу «Users» и затем, происходит подключение

После этого, создаём класс-модель для работы с нашей таблицей

Затем, создаём класс для работы с базой данных

Следующим шагом мы будем добавлять объект в БД. Для этого прописываем следующий код (Рис. 23)

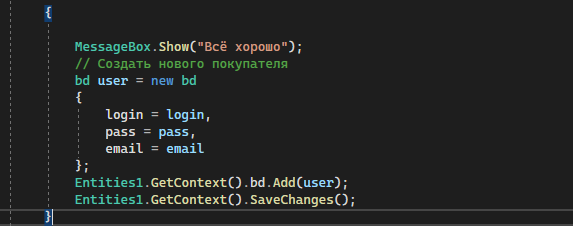


Рисунок 24. Код для введения данных в базу данных

После того, как мы добавили данные в нашу таблицу, нам необходимо отобразить наши данные.

После проверки нашей программы, переходим к окну с авторизацией. Создаём новое окно AuthWindow.xaml, в файле AuthWindow.xaml.cs прописываем код стиля, после чего получается следующее окно.(рис .25)

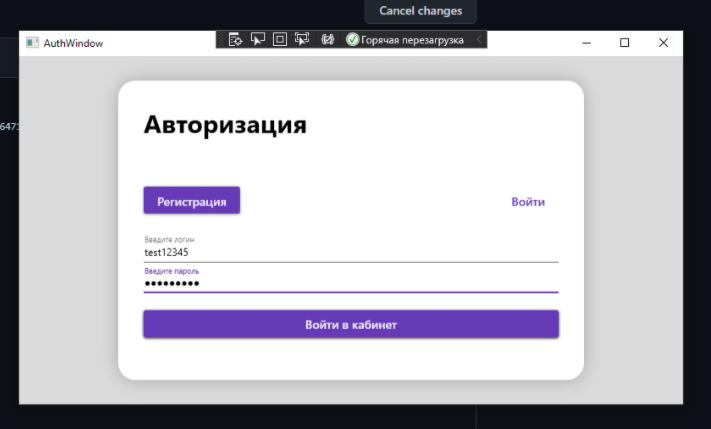


Рисунок 25. Окно авторизации

Затем, добавим проверку введённых пользователем данных и сверение их с данными из нашей Базы данных (Рис. 26)

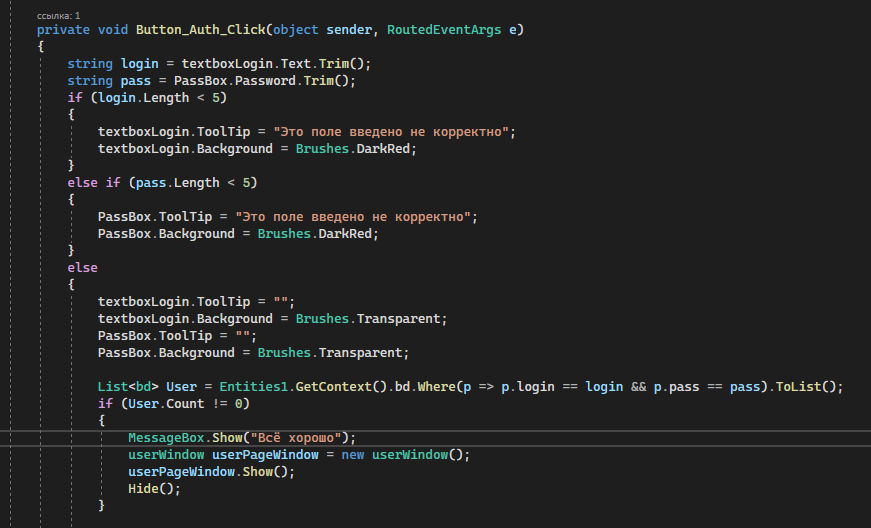


Рисунок 26. Проверка введённых пользователем данных и сверение их с данными из нашей Базы данных

После написания кода, проверяем работу нашей программы (Рис. 27)



Рисунок 27. Проверка нового пользователя

После создания окна авторизации, можно перейти к созданию «Кабинета пользователя». Для этого нам необходимо настроить переход между страницами (Рис. 28).

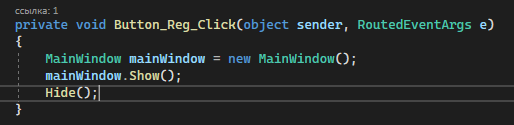


Рисунок 28. Код для перехода между страницами

Далее, создадим само окно «Кабинет пользователя» (Рис. 29).

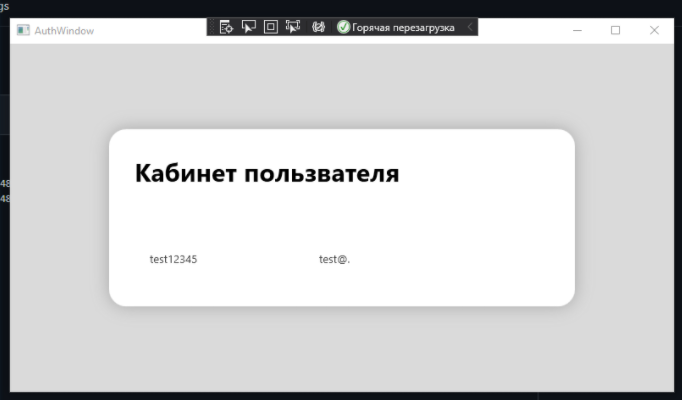


Рисунок 29 Кабинет пользователя

Прописываем код вывода всех зарегистрированных пользователей (Рис. 29) и проверяем работу программы (Рис. 30).

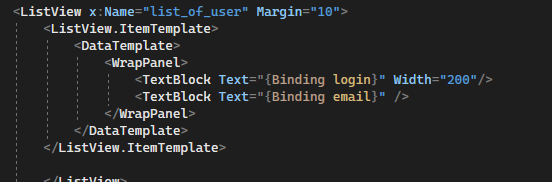


Рисунок 30. Код вывода всех зарегистрированных пользователей

Вот мы и закончили создание нашего приложения WPF. Во время разработки этого приложения, мы научились работать с базой данных, создавать переход между окнами, добавлять новых пользователей в базу данных и т.д. Разработка данного приложения нужна была для закрепления навыков создания подобных приложений и работы с базой данных.

## 2.4 Нейросети

Сначала мы подключаем библиотеки OpenCV, face\_recognition b dlib. Как это сделать описано выше

Далее написание кода (Рис. 31)

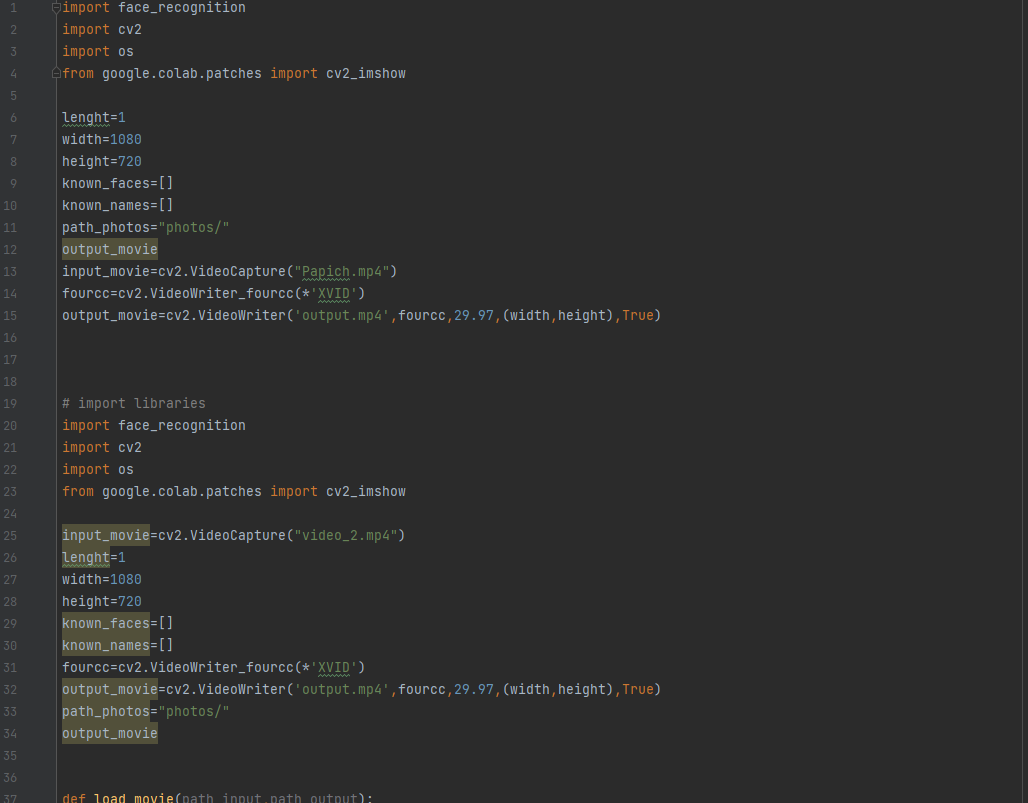


Рисунок 31. Написание кода



Рисунок 33. Обнаружение жеста Бумага

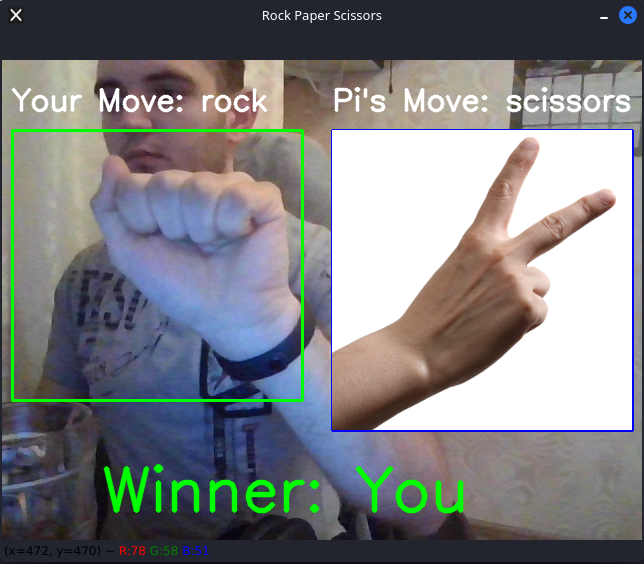


Рисунок 34. Обнаружение жеста Камень.

**Выводы**

По окончании учебной практики были решены следующие задачи:

1. Знакомство с литературой
2. Изучить библиотеку OpenCV
3. Реализация приложения на WPF

В ходе этой практики были сформированы следующие профессиональные компетенции:

ПК 1.1 Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.

ПК 1.2 Разрабатывать программные модули в соответствии с техническим заданием.

ПК 1.3 Выполнять отладку программных модулей с использованием специализированных программных средств

ПК 1.4 Выполнять тестирование программных модулей

ПК 1.5 Осуществлять рефакторинг и оптимизацию программного кода

ПК 1.6 Разрабатывать модули программного обеспечения для мобильных платформ.

ПК 2.1. Разрабатывать требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент.

ПК 2.2. Выполнять интеграцию модулей в программное обеспечение.

ПК 2.3. Выполнять отладку программного модуля с использованием специализированных программных средств.

ПК 2.4  Осуществлять разработку тестовых наборов и тестовых сценариев для программного обеспечения.

ПК 2.5 Производить инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования.

# Список литературы

1. [Орлов С.А.](http://library.sgu.ru/cgi-bin/irbis64r_15/cgiirbis_64.exe?LNG&amp;Z21ID&amp;I21DBN=NIKA&amp;P21DBN=NIKA&amp;S21STN=1&amp;S21REF=1&amp;S21FMT=fullwebr&amp;C21COM=S&amp;S21CNR=20&amp;S21P01=0&amp;S21P02=1&amp;S21P03=A%3D&amp;S21STR=%D0%9E%D1%80%D0%BB%D0%BE%D0%B2%2C%20%D0%A1%D0%B5%D1%80%D0%B3%D0%B5%D0%B9%20%D0%90%D0%BB%D0%B5%D0%BA%D1%81%D0%B0%D0%BD%D0%B4%D1%80%D0%BE%D0%B2%D0%B8%D1%87) Программная инженерия. Технологии разработки программного обеспечения: учебник / С. А. Орлов. – 5-е изд., обновл. и доп. – Москва ; Санкт- Петербург:Питер, 2016. – 640 с.
2. Джепикс Филипп, Троелсен Эндрю. Язык программирования C# 7 и платформы .NET и .NET Core / Джепикс Филипп, Троелсен Эндрю – М.; Вильямс, 2018. – 1328. «Наука», 2017 – 87с.
3. Павловская Т.А. С/С++. Программирование на языке высокого уровня. СПб.: Питер, 2017. – 461 с.
4. Семакин И. Г. Основы алгоритмизации и программирования : учебник для студ. учреждений сред. проф. образования / И. Г. Се- макин, А. П. Шестаков. — М. : Издательский центр «Академия», 2017. — 304 с
5. Хабибулин И.Ш. Программирование на языке высокого уровня. C/C++. – СПб.: БХВ– Петербург, 2017. – 512 с.
6. Прайс Марк. C# 7 и .NET Core. Кросс-платформенная разработка для профессионалов / Прайс Марк – Москва; Вильямс, 2018. – 604.
7. Скит Джон. C# для профессионалов. Тонкости программирования / Скит Джон – Москва; Вильямс, 2019. – 608.
8. Мюллер Д. П. C# для чайников / Мюллер Д. П. – Москва; Москва, 2019, – 608 с.
9. [Главная | GeekBrains - образовательный портал (gb.ru)](https://gb.ru/)
10. https://opencv.org/courses/
11. <https://robocraft.ru/page/opencv/>
12. Skillbox: <https://live.skillbox.ru/playlists/code/python/>
13. https://github.com/vladimir75vov/Vladimir\_Budaev\_195/blob/main/Practice/sourse/%D0%94%D0%9D%D0%95%D0%92%D0%9D%D0%98%D0%9A\_%D0%A3%D0%9F.01.docx