|  |
| --- |
| VILNIAUS KOLEGIJA/UNIVERSITY OF APPLIED SCIENCES  FACULTY OF ELECTRONICS AND INFORMATICS  SOFTWARE ENGINEERING DEPARTMENT |
|  |
| **Face recognition** |
| **Project** PI20E |
| STUDENTS Hlib Fediun |
| (SIGNATURE)  2022-11-29 |
| LECTURER lect. A. Savickaitė |
| (SIGNATURE) 2022-11-29 |
|  |
| 2022 |

# TABLE OF CONTENTS

[TABLE OF CONTENTS 2](#_Toc121783373)

[TABLE OF PICTURES 2](#_Toc121783374)

[INTRODUCTION 3](#_Toc121783375)

[SOFTWARE IMPLEMENTATION 4](#_Toc121783376)

[CONCLUSIONS 8](#_Toc121783377)

# TABLE OF PICTURES

[Picture 1. Python and OpenCV library 3](file:///C:\Users\Hlib%20Fediun\OneDrive%20-%20Vilniaus%20kolegija\Desktop\Face%20recognition\Face_recognitio%20project.docx#_Toc121783321)

[Picture 2. Hand Tracking program 3](#_Toc121783322)

[Picture 3. Installation of Python 4](#_Toc121783323)

[Picture 4. OpenCV installation 4](#_Toc121783324)

[Picture 5. MediaPipe installation 4](#_Toc121783325)

[Picture 6. Import of OpenCV 5](#_Toc121783326)

[Picture 7. Structure 5](#_Toc121783327)

[Picture 8. Code of the program 5](#_Toc121783328)

[Picture 9. Code review 6](#_Toc121783329)

[Picture 10. Code review 6](#_Toc121783330)

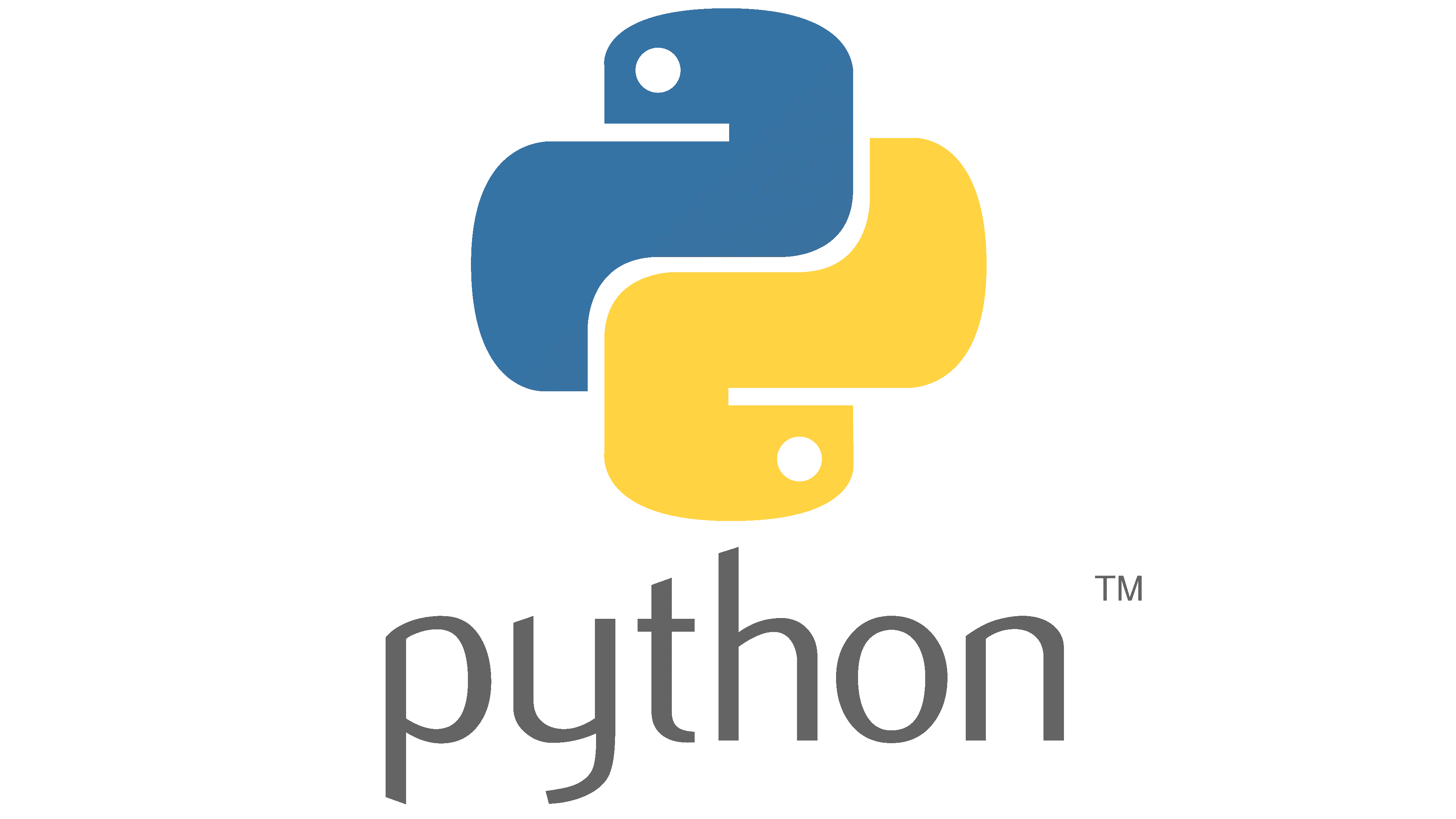
[Picture 11. Code review 6](#_Toc121783331)

[Picture 12. Result cases 7](file:///C:\Users\Hlib%20Fediun\OneDrive%20-%20Vilniaus%20kolegija\Desktop\Face%20recognition\Face_recognitio%20project.docx#_Toc121783332)

# INTRODUCTION

This project is a final work of face recognition course. The goal of it to create the application using the python programming language with an open source computer vision and machine learning software library called OpenCV. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being an Apache 2 licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

Picture 1. Python and OpenCV library



To perform this project, working process was split on three parts:

1. Prototyping the project;
2. Set up and implementing in code ;
3. Improvements and final documentation with presentation;

The project is an implementation of Hand Tracking program created using OpenCV functionality. Basically the program allows you to calculate your fingers on your hand through the looking on a real time camera stream. The code of the program divides your hand on some landmark of connected between each other dots and calculate how many fingers you are showing now. At the same time, you can show only one hand, because the code is not optimized for calculating mor than 5 times, however, the second hand will also divide by landmark of dots. All these staff was implemented using python programming language and OpenCV library. Technical information about how exactly the game works is provided in following section of the document.

A person wearing headphones

Description automatically generated with low confidence

Picture . Hand Tracking program

On this picture we could first appearance of our Hand Tracking program.

# SOFTWARE IMPLEMENTATION

This project is created using Python programming language and its library for face recognition staff called OpenCV. Prior to make any explanations of how it works, the steps of its installation will be presented.

Installation process is super-fast and don’t require any additional command other than installation one. After installation is complete you able to use all framework parts in any of your project.

Graphical user interface, text

Description automatically generated

Picture . Installation of Python

To install and get access to OpenCV library and it’s functions you need to open terminal in Visual Studio code(or in other redactor) and write the “*pip install opencv-python*”. Later on after successful installation you will see message like this:

Text

Description automatically generated

Picture . OpenCV installation

Also, we need to install MediaPipe framework. MediaPipe is a cross-platform pipeline framework to build custom machine learning solutions for live and streaming media. The framework was open sourced by Google and is currently in the alpha stage. To make this we need to write in terminal *pip install mediapipe*.

Text

Description automatically generated

Picture . MediaPipe installation

After that you just need to write down the *import cv2* and print statement to check the version of the downloaded library, and at all if it installed correctly.

Graphical user interface, text, application

Description automatically generated

Picture . Import of OpenCV

After all language and framework parts we can begin with the code discussion. The program is consisting of single file *main.py,* where written all the functionality and logic of our program Hand Tracking.

Text

Description automatically generated

Picture . Structure

Text

Description automatically generated

Picture . Code of the program

On this picture we could see whole code of our Hand Tracking application.

* Lets start our code review. So, the first two line we need for importing functions from OpenCV and MediaPipe frameworks, that we installed before. For this statement we wrote *import.*
* In the second part we initialized hand tracking modules, this functions we wrote in 4-8 lines. We created the variables there and putted in them MediaPipe solution statements, so that our program could recognize our hands on the real time camera.
* The next statement we wrote for capturing the video from our laptop camera *cap = cv2.VideoCapture(0)*.
* Then we checking if the camera opened or no, inserting functions for reading frame from camera, converting image from BGR to RGB, processing Image for Tracking. All these thing we wrote in this part from 13 to 19 lines of code:

Text

Description automatically generated

Picture . Code review

* The next part explains the main logic of our Hand Tracking program. In this part we drawing landmark of connected with each other dots on our hands, hand connections, logic that will calculate how many fingers we are showing now on camera of our laptop and drawing the number that represents number of fingers that we are showing. We wrote it from 21 to 37 lines of code.

Text

Description automatically generated

Picture . Code review

* The last two lines we wrote for possibility of closing the program by pressing the “Q” on the keyboard, if the break statement is not working.

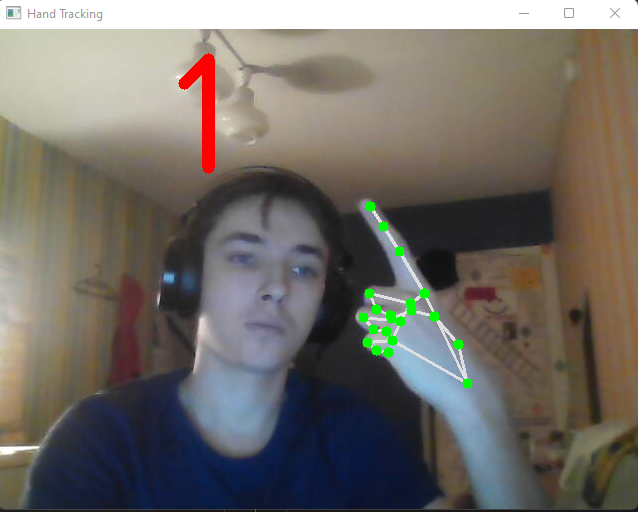
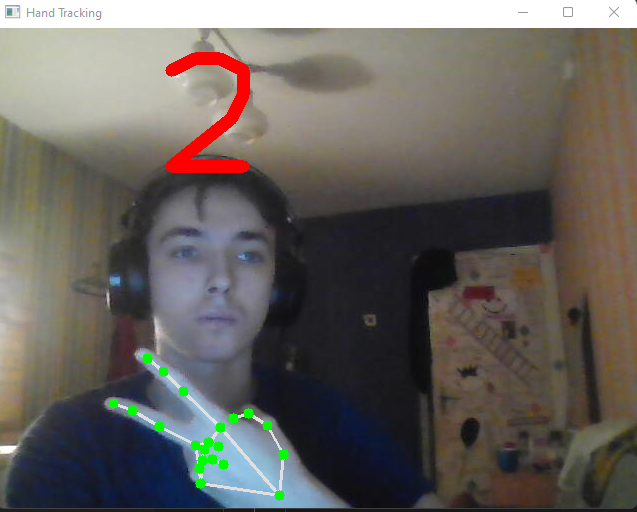
Text

Description automatically generated

Picture . Code review

Finally, in the result, we get several cases in which program calculating how many fingers I am showing in the real time:

Picture 12. Result cases



On the first image we see that I an showing one finger and program drawing number one, and the same algorithm works for two, three, four and five fingers. Also we see landmark of connected dots on my hand that we created due to Mediapipe framework.

# CONCLUSIONS

* The final project for the Face Recognition course is completed and described. A working process started from the formulating of task and selection of software tools. After learning a basics of Python language a OpenCV framework was selected as a base tool for creating Hand Tracking program as project work.
* As the result of the work on this project a lot about basics of Python have been learned and skills in writing basic constructions on this language were improved. The result is combination of basic Python constructions with OpenCV modules.
* Knowledge got on this course will be applicable in future projects as well as will help better understand different approaches to programming. Course also introduced with such terms as high level, dynamically typed and interpreted language, showing its meaning on practical implementation with Python.
* All the task that mentioned in introduction part was completed with no problems. Idea we took from learning video course of famous teacher from Python language with OpenCV framework(prototyping). Set up and implementation took no more than couple of days, for this we used Visual Studio Code redactor. For implementation we did counter that calculates fingers on the screen. And documentation we created in Microsoft Word application for a one day.
* Vision on a future: this application we could connect with face recognition field in some way. For example, we could add side detecting, so the program could now where we showing finger. If the big finger showing up, so we have good mood, if down, so we are sad. Or for example, we are showing two fingers and one eye is closed, so we are funny and want to take a picture. This application we can use as a mask in social media networks, as a: Snapchat, Instagram etc., for emotion detection.