Faculty of Engineering Helwan University

4th Year Computer Department

Computer Vision (Dr.Motasem Elshourbagy)

- **Project Name: Smart Face Identifier & Styler**
- **Group Number: 2**
- **Submission Date: 14 / 6 / 2022**

- **Project Administrators**

1- Noor El-Deen Magdy Mohamed Sec (4)

4- Hady Mohamed Kamel Sec (4)

2- Ali Fathy Abbas Sec (3)

3- Rania Mostafa Sec (2)

- Table of Contents

Project Name: Smart Face Identifier & Styler 1

Project Administrators 1

Table of Contents 2

Problem Description: 3

Proposed Solution: 5

2.1 Block diagram (diagram for your system) 5

Functions Description 6

2.2 Implementation Tools and Steps 6

Implementation Tools 6

Functional Requirements 7

Non-functional Requirements 8

Use Cases Diagram 8

How to use the application: 10

How the work is distributed among team members: 11

1. Problem Description:

- We need a system or a project about using **cameras from the desktop computer or a laptop** so it captures photos and has many other features like the following:

- **Capturing** and **saving photos** from live capture to a gallery, which includes opening the laptop camera or using an external camera device.

- The application can also **identify the human** so it can say for example the application says that the person interacting with it is human and displays his/her name above him so it's a good way to not forget the name of the person we talk to.

- If the application detects that the user is not known for it, it automatically prompts him/her to ask if they want to be known later by the application and **save his/her picture for further identification,** and then upon using the camera again the application detects the face, displays the name and give full control of the other features, otherwise, the application will terminate as it does not know have a database entry of the person using it.

- Another feature the application needs to do is to apply **filters or style transfer** (no camera application without applying filters, no?), So when the application detects there's a face the application gives the user the ability to change the layer over his face, for example, the layer over the human face is normally a rectangle but when using this feature the user can change it to anything he/she likes, also the application can take a photo from the user and a photo that he/she wishes to merge together to add a **custom effect or its style** to his/her photo

- Another feature this application needs to have, imagine that there's an online meeting for some students and the teacher wants to write notes but its too hard to write using the mouse and it makes the words look harder to read and understand, so what about we make it like when the camera application detects that there's a pencil object it prompts the user to choose the color of his desire then he/she can use that color and the **pencil to write on the screen** so it is visible for the students to read and understand.

2. Proposed Solution:

2.1 Block diagram (diagram for your system)

- The system implementation is described below in the following diagrams for a more detailed view of the project development.

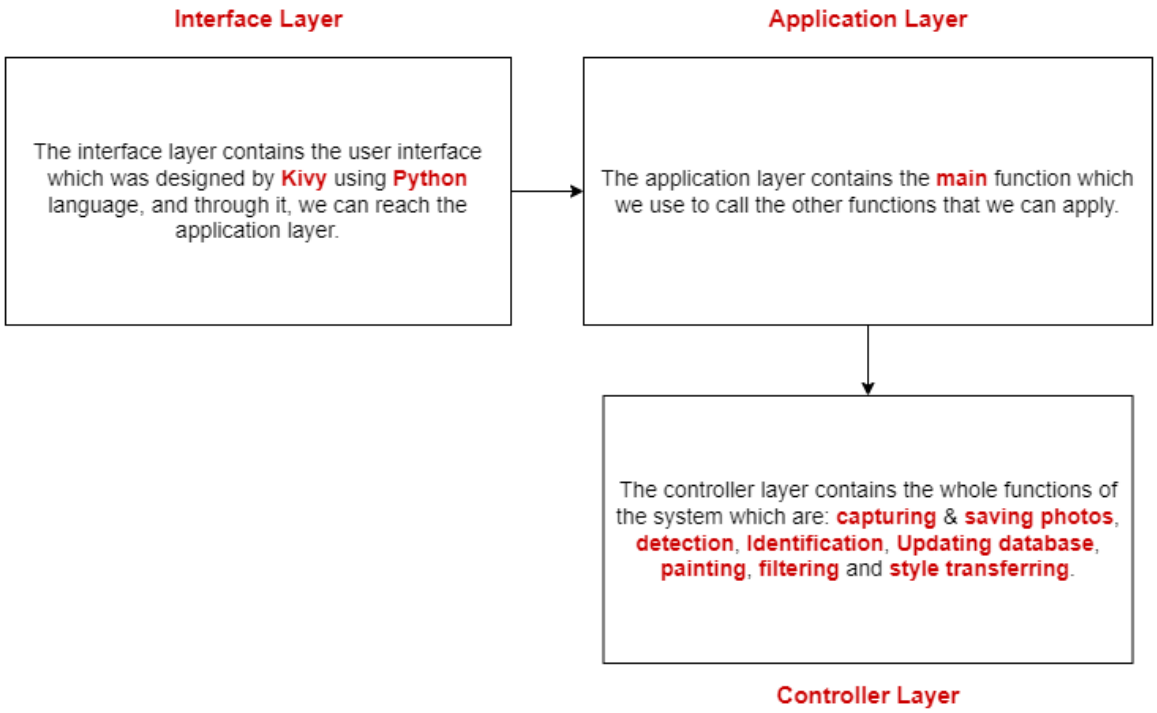

Fig 1. System Block Diagram

- **Functions Description**

- **Capturing** and **saving photos**.
- **Detection:** Detect the human face
- **Identification:** Identify the human name
- **Update Identifier:** save his/her picture in the faces pickle file for further identification
- **Painter:** We use a pencil to write on the screen
- **Image Segmentation Filter:** Apply image segmentation filter
- **Style transfer**: Can take a photo from the user and a photo that he/she wishes to merge together to add a **custom effect or its style** to his/her photo
- **Main function:** Open the camera then use the previous functions

2.2 Implementation Tools and Steps

- Our solution in this project aims to make a smart camera app that will allow the user to make personal identification, detection, paint on the camera, and more.

1. Implementation Tools

The Project environment can be summarized into 3 Parts	
1- Main programming language.	**Project as whole was written in Python 3.**
2- Logic.	**Project logic is mainly based on OpenCV 2 library in Python.**
3- GUI.	**Project GUI was made with Kivy framework in Python.**

Table 1. Project Tools

2. Functional Requirements

- Some of the constraints and limitations were the requirement of a good quality camera on the laptop or PC, Clear person photos for face extraction.

- Our proposed features & benefits are summarized below in the following functional and non-functional requirements shown for an abstract and detailed view of the app development goals.

REQ - x	Description
REQ - 1	Object detection when using the camera
REQ - 2	Faces identification when using the camera
REQ - 3	Saving the face of the user in the gallery for further identification
REQ - 4	Applying style transfer or filters for any photo
REQ - 5	When detecting a pencil/pen the application prompts the user to choose a color from the screen to start to write over the screen
REQ - 6	A Gallery to add the Saved Faces and filters to
REQ - 7	Capturing photo to save it in the gallery

Table 2. Functional Requirements

3. Non-functional Requirements

REQ - x	Description
REQ - 8	The requirement of a good quality camera in the laptop or PC.
REQ - 9	Clear person photos for face extraction

Table 3. Non- Functional Requirements

4. Use Cases Diagram

I. Use Case Description

- **The Actor (User)**
- Stakeholders will be users that want to detect faces like in crime-fighting, in social media applications, and in some online teaching sessions
- **There are mainly 1 function (Live Camera)**, however, the user can only start the application using the live camera so the user opens their camera, and then he/she can use any of the 6 features of the application listed in the diagram.
- **Save Picture:** when a user opens the application for the first time this function is triggered to save his face/picture.
- **Detect Human:** returns a rectangle shape over the user's face.
- **Identify:** returns the name of the user IF he/she is found in the database.
- **Update gallery:** updates the gallery when a new face is added to it.
- **Painter:** uses a pen object to draw over the screen.
- **Style Transfer:** merges/adds filters with the user photo/life.

- **Browse:** browses the gallery and behave like the live capture except for it does not require a camera to work.

II. Use Case Description

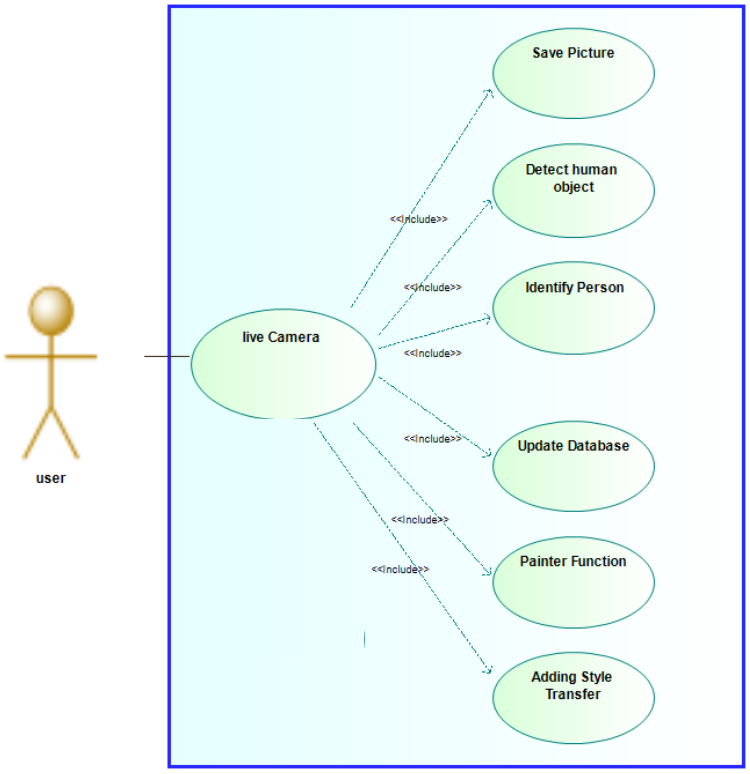

Fig 2. Use Case Diagram

3. How to use the application:

1. Open cmd in the application directory and type "**pip install -r requirements.txt**" to install the required dependencies to run the application, then type "**python main.py**" to run it.
2. It opens on the live camera and we can see different features on the left.
3. If we want to identify the man, Firstly, we choose "**Detect Faces**" to draw a rectangle around the face, then press "**Identify Faces**" which tells us who this man is by comparing the face with the faces that were saved in the faces directory.
4. Press "**Image Segmentation**" if you want to divide the image into its components (edge detection).
5. It's simple if you want to write on the screen by pressing the "**Painter**" button, then choose which color you want to write with, but we need to use any object that has a blue color end.
6. "**Capture Screen**" to save a picture from the stream in the gallery directory.
7. "**Capture Video**" to save a video from a stream in the gallery directory after pressing the "**q**" button on the keyboard.
8. To merge between two images one from the live stream and the second from saved styles, press "**Style Transfer**". From the keyboard, if we
    A. Press "**n**" to load the next neural style transfer model.
    B. Press "**p**" to load the previous neural style transfer model.
    C. Press "**s**" to save the resulting image to the gallery directory.
9. "**Clear**" button to delete any feature we have used.
10. To add a new person we should first create a directory manually with his name and add his picture, then press "**Update Identifier**".
11. Press "**Esc**" on the keyboard to exit from the application.

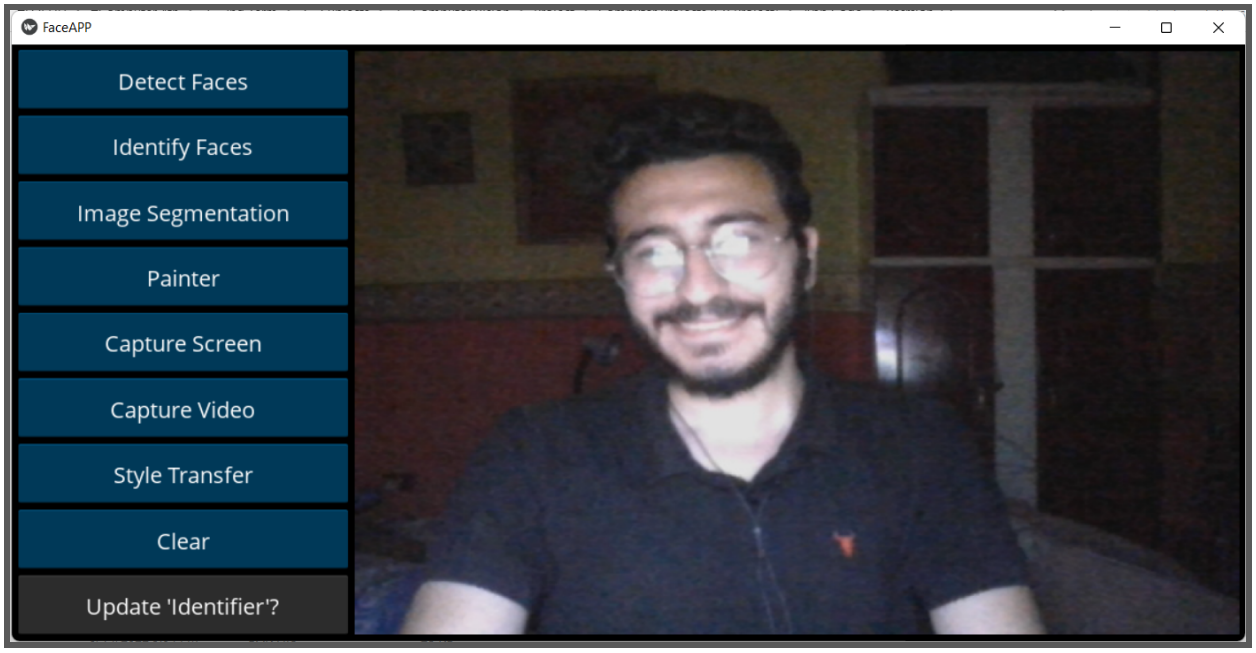

Fig 3. A screenshot from the App

4. How the work is distributed among team members:

Student Name	Contributions
Noor El Deen Magdy Mohamed	Developed the python core version of the app and the face identifier.
Ali Fathy Abbas	Worked on style transfer capture and save camera output features.
Hady Mohamed Kamel	Worked on the face detector and style transfer features.
Rania Mostafa	Worked on the image segmentation and painter features.

Table 4. Team Contributions