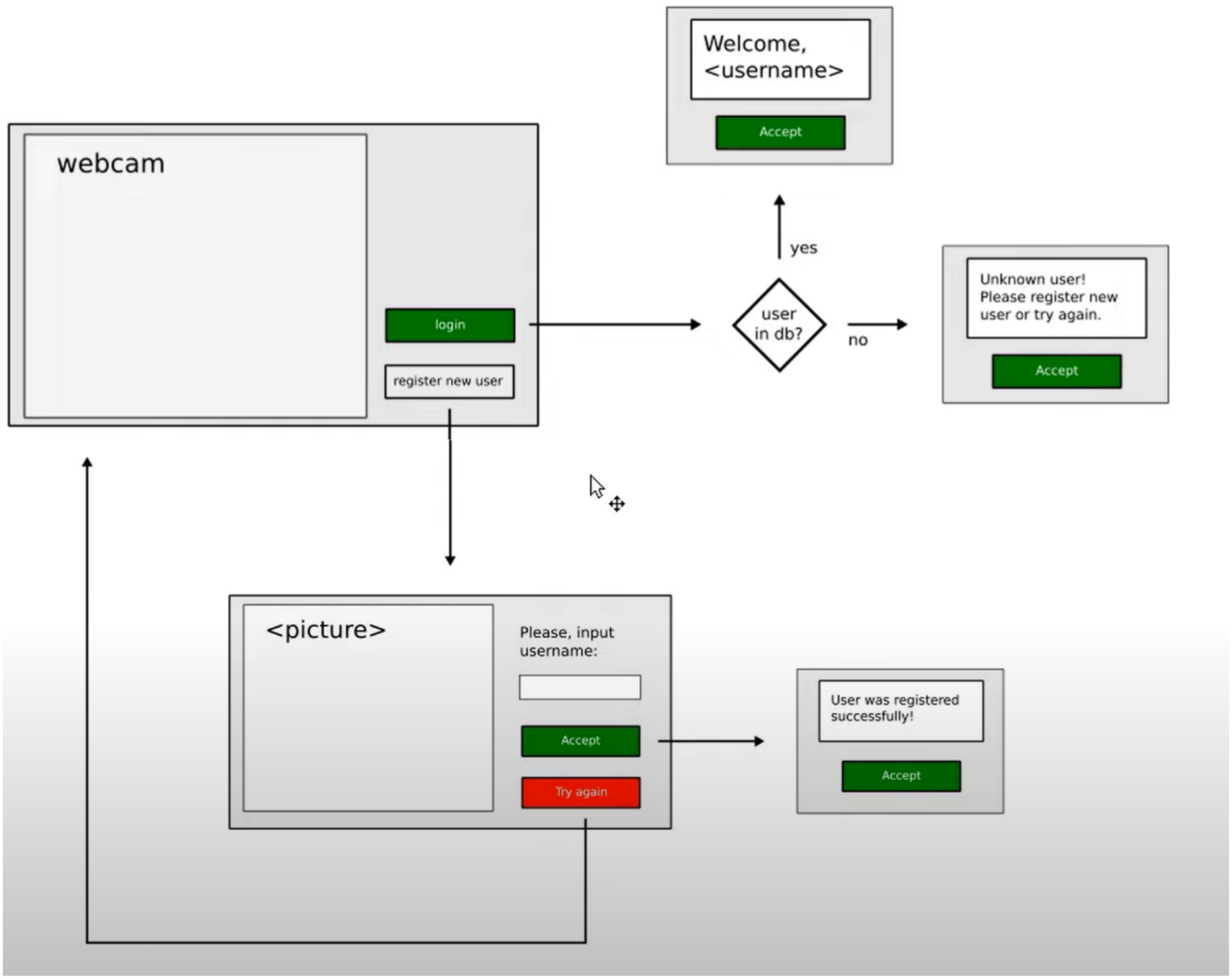
Part 3 : High level design of the proposed solution

Submitted by
Yessine Barkia
Roaya Neffeti

User Interface :



✅ The system shall provide the following functionalities. (subject to change)

✅ The logging in and registration steps are somewhat similar. They both require the image preprocessing, face detection, and feature extraction steps. The only difference is:

➕ In registration, the face is saved in the database

➕ In logging in, the features are compared with each face in the database.

✅ The following diagrams propose the solution that we will deploy in this face recognition system for start to finish. (subject to change)

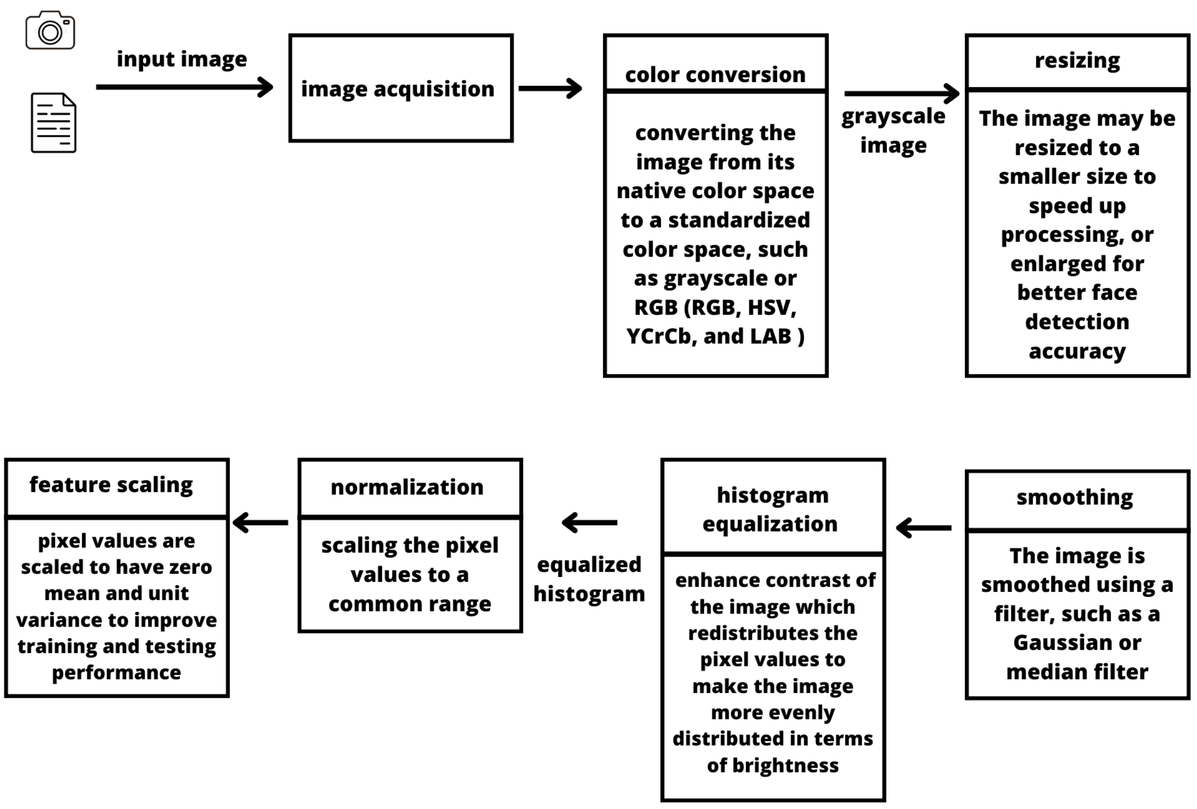
Image preprocessing :

Goal : to enhance the quality of the image, making it easier for the face detection algorithm to accurately locate and identify faces within the image

→ The order of the image preprocessing steps in OpenCV can depend on the specific requirements of the task at hand and not all of these steps are required for every application.

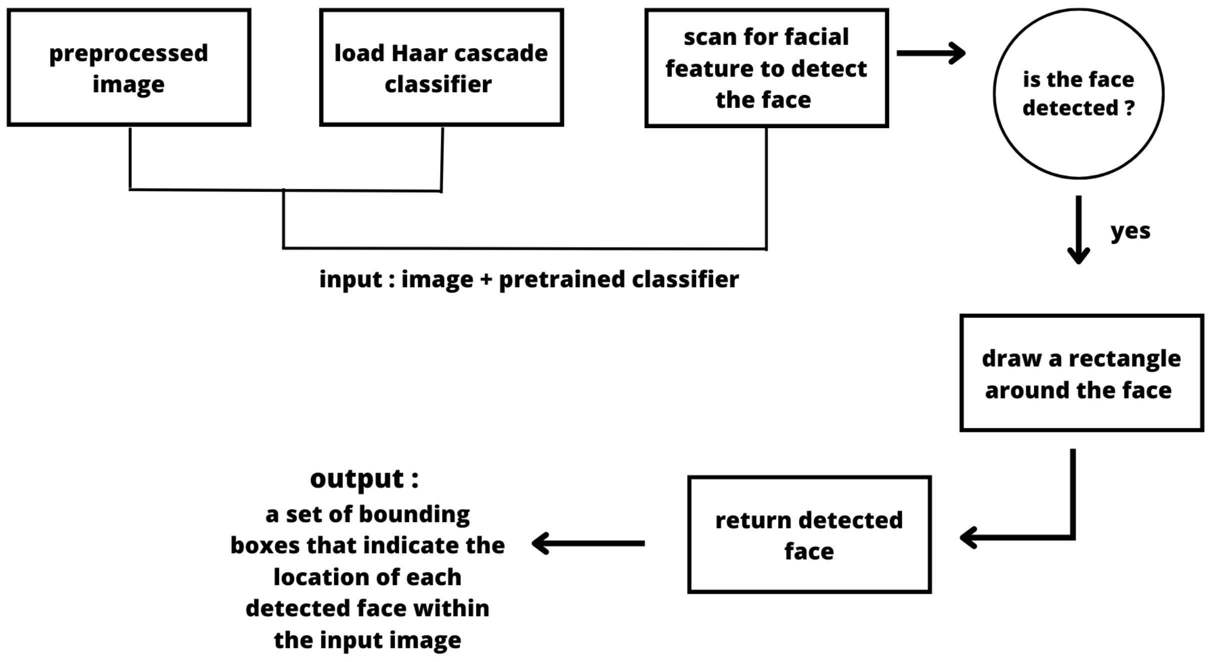
The common steps are :

1. Image acquisition: capturing or loading the image data into memory
2. Color conversion: converting the image from its native color space to a standardized color space, such as grayscale or RGB
3. Resizing: resizing the image to a standard size, if necessary
4. Smoothing: applying a filter to the image to reduce noise or blur
5. Histogram equalization: enhancing the contrast of the image by adjusting the intensity distribution
6. Normalization: scaling the pixel values to a common range, if necessary
7. Feature scaling : If the image will be used as input to a machine learning algorithm


input image
image acquisition
color conversion
converting the image from its native color space to a standardized color space, such as grayscale or RGB (RGB, HSV, YCrCb, and LAB)
grayscale image
resizing
The image may be resized to a smaller size to speed up processing, or enlarged for better face detection accuracy
smoothing
The image is smoothed using a filter, such as a Gaussian or median filter
histogram equalization
enhance contrast of the image which redistributes the pixel values to make the image more evenly distributed in terms of brightness
equalized histogram
normalization
scaling the pixel values to a common range
feature scaling
pixel values are scaled to have zero mean and unit variance to improve training and testing performance

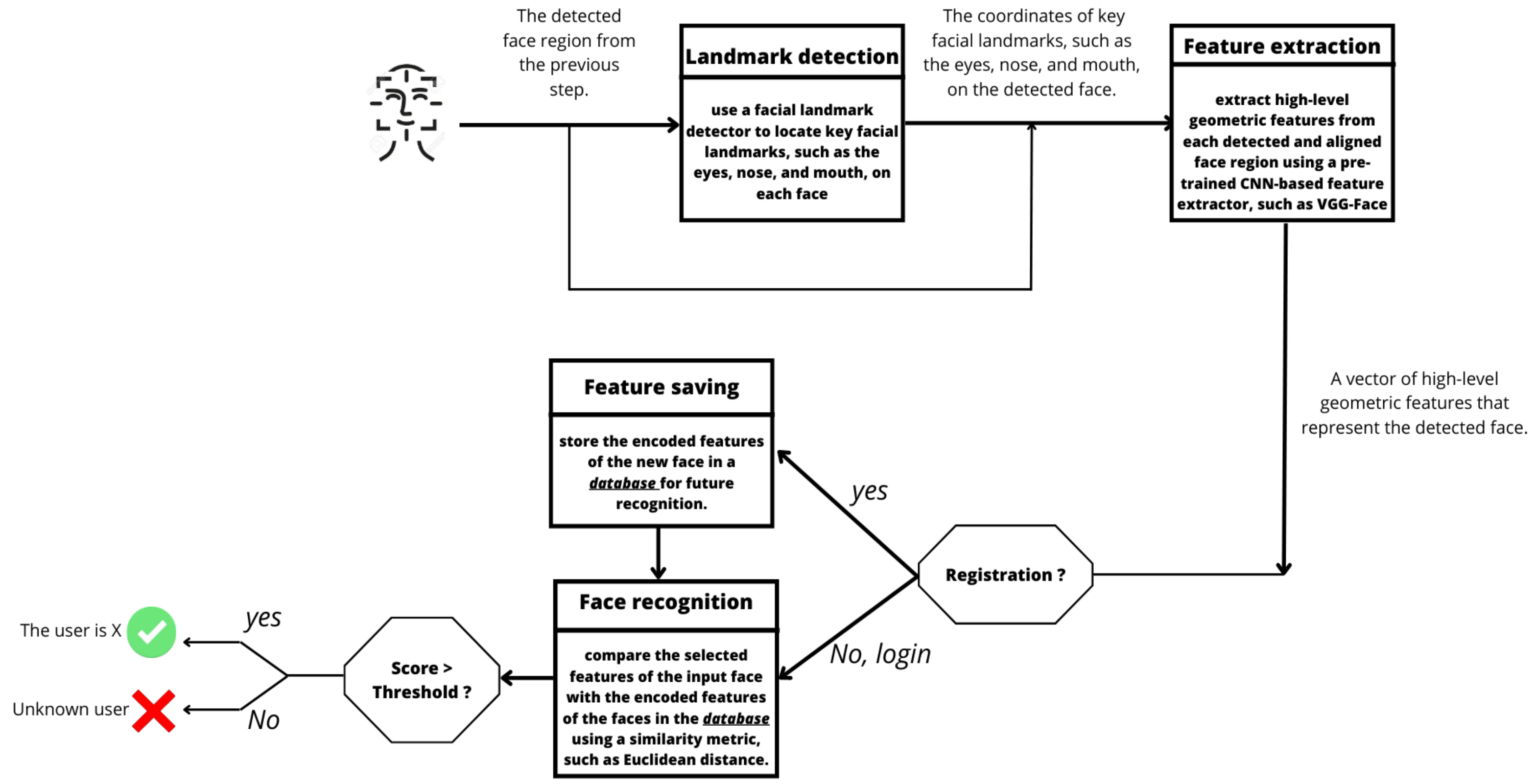
Face detection :

Once the image has been preprocessed, it can be passed to the face detection algorithm, which will analyze the image and attempt to locate regions that contain human faces. This process may involve using various techniques such as feature detection or machine learning-based classifiers to identify facial features within the image
→ The algorithm that we will be using for the face detection phase is the Haar cascade classifier
Haar Cascade Classifier : an Object Detection Algorithm used to identify faces in an image or a real time video


preprocessed image
load Haar cascade classifier
scan for facial feature to detect the face
is the face detected ?
yes
input : image + pretrained classifier
draw a rectangle around the face
return detected face
output :
a set of bounding boxes that indicate the location of each detected face within the input image

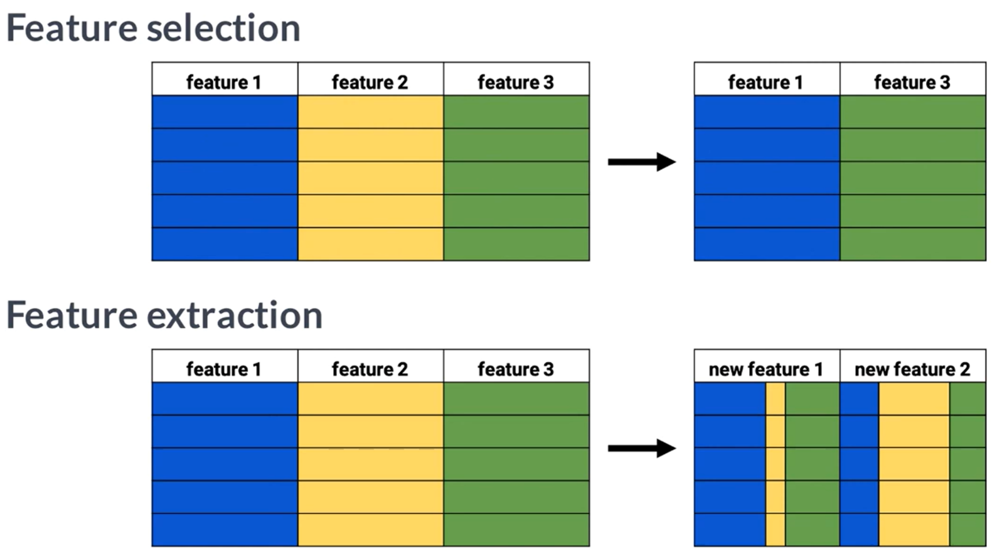
→ The loaded haar cascade classifier file is an Adaboost cascaded face detection classifier based on Haar features. It is trained by extracting feature information from a large amount of face image information ;It is essentially a set of rules that define what a face looks like and how it can be distinguished from other objects in an image.

Input / Output messages :



The detected face region from the previous step.
Landmark detection
use a facial landmark detector to locate key facial landmarks, such as the eyes, nose, and mouth, on each face
The coordinates of key facial landmarks, such as the eyes, nose, and mouth, on the detected face.
Feature extraction
extract high-level geometric features from each detected and aligned face region using a pre-trained CNN-based feature extractor, such as VGG-Face
A vector of high-level geometric features that represent the detected face.
Registration ?
yes
Feature saving
store the encoded features of the new face in a database for future recognition.
No, login
Face recognition
compare the selected features of the input face with the encoded features of the faces in the database using a similarity metric, such as Euclidean distance.
Score > Threshold ?
yes
The user is X
No
Unknown user

<u>Side note</u>: *Compared to feature selection, feature extraction is a completely different approach, but with the same goal of reducing dimensionality. Instead of selecting a subset of features from our data set, we'll be calculating or "extracting" new features from the original ones. These new features have as little redundant information in them as possible, and are therefore fewer in number. One downside is that the newly created features are often less intuitive to understand and original ones.*


Feature selection
feature 1
feature 2
feature 3
feature 1
feature 3
Feature extraction
feature 1
feature 2
feature 3
new feature 1
new feature 2