A documentation of the

**Face Detection and**

**Quick Response Code Scanner System**

A partial fulfillment of the requirements for the midterm of first semester in the course Modelling and Simulation

**Polytechnic University of the Philippines**

**Department of Computer and Information Sciences**

**BSCS 2-3**

**Group 4**

**Table of Contents**

|  |  |
| --- | --- |
| Table of Contents | 2 |
| Member Roles | 3 |
| Project Description | 3 |
| System Snapshot | 4 |
| System Pseudocode and Flowchart | 7 |
| Libraries | 11 |
|  |  |
| Project Features Source Code (Back-end) | 12 |
| Face Detection | 12 |
| Face Recognition | 14 |
|  |  |
| Addendum | 16 |

**Members and Developers**

**Project Manager and System Developer**

Calendario, Mark Kenneth

**User Interface and User Experience Designers**

Abarre, Jeo

Sinalubong, Sean Matthew

**System Analyst**

Sinalubong, Sean Matthew

**Business Analyst**

Cruz, Jan Miles

**Technical Designers**

Gani, Zaimmon

Yim, Gwyneth Ann Marie

**Technical Writers**

Brigola, Jaycee

Vedasto, Micole Aaron

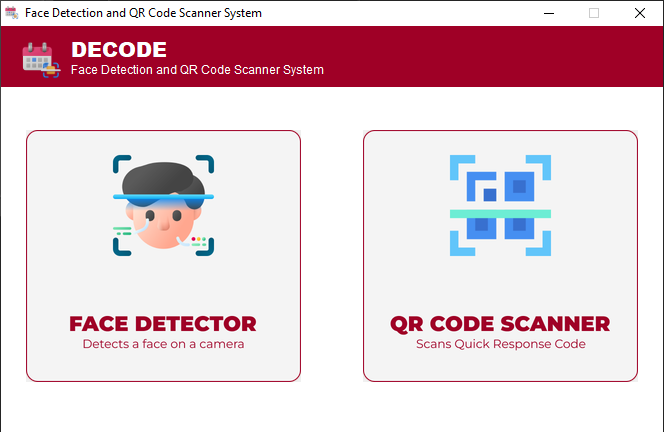
**Project Description**

The QR Code Scanner and Face Detection System is a combination of two useful features: quick response code scanning and decoding and frontal face detection. The QR code scanning feature allows users to decode and get information encoded in a machine-readable black and white matrix by simply pointing their device's camera at a QR code, while the face detection feature uses advanced computer vision algorithms to detect and track faces in real-time. This project is designed to be user-friendly, with a simple and intuitive interface.

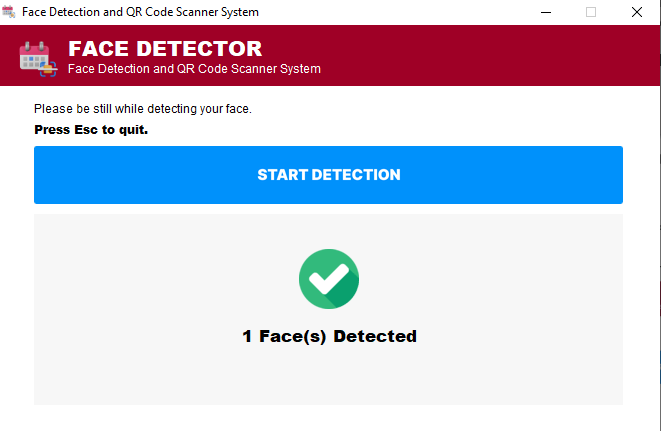
**System Snapshot**

**Main Frame of the System**

Upon launching the application by clicking on the executable file (exe), the main frame of the application will appear. The root layer of the graphical user interface contains two buttons for the main features of the system: a face detector and a QR code scanner.



**Face Detection Data Frame**

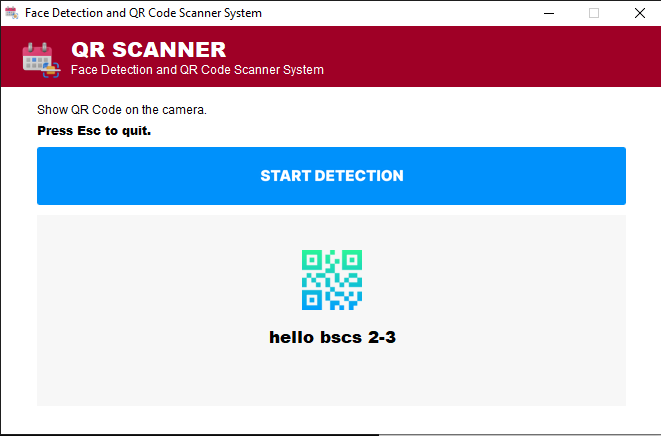
This Face Detector frame is stackable from the root layer of the GUI. It contains the control for starting the detection and the guides on how to use and exit the frame. This frame is responsible for displaying the real-time state of the camera device, face detection, and faces count on the frame. The start detection button is placed on this frame for starting up a real-time camera stream wherein the face detection will take place. A picture containing person, indoor

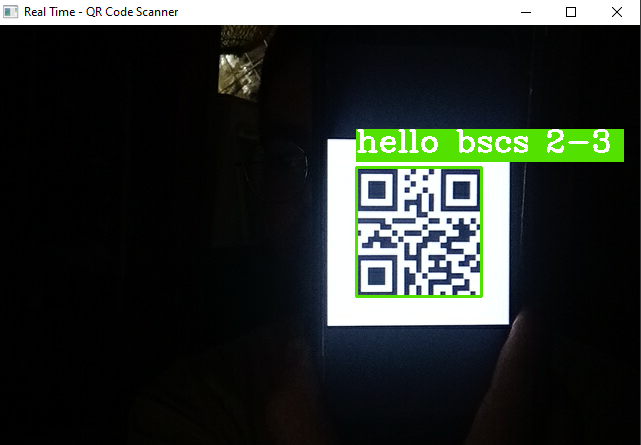
Description automatically generated

**QR Code Scanner Data Frame**

This QR Code Scanner Data Frame is also stackable from the root layer. Once the QR Code Scanner button was clicked from the main frame, the frame below will popup. This frame consists of real-time data of decoded QR code and other indicators from the camera stream.

The start detection button is placed and packed on this frame for showing up a real-time camera window for the QR Code scanning to take place.





**System Pseudocode and Flowchart**

This portion of this paper will discuss and show the flowchart or diagram representation of the system and pseudocode or flow of the system Face Detection and Quick Response Code Scanner System in text format. The flowchart and pseudocode of the system is divided into three parts:

1. Main Panel of the System  
 2. Face Detection Data Frame  
 3. QR Code Scanner Data Frame

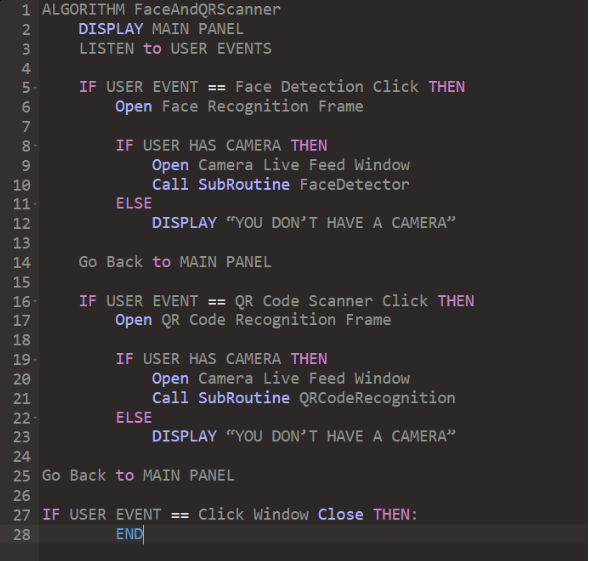
**Root Layer of the App**

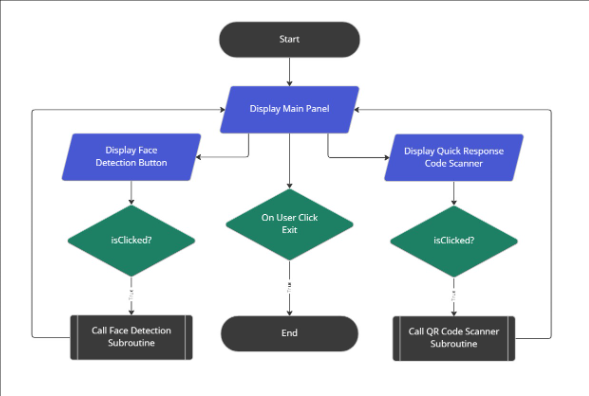
This algorithm shows how the root layer of the application will work. First, it initializes the GUI and display it on the computer screen. Then, it will wait to the user click events.

If user clicked the face detection button, the face detection frame will pop up. After that, the system will check if the user has a camera, if so, it will call the Face Detector subroutine to start the face detection. Otherwise, it will display “you don’t have a camera”.

However, if user clicked the QR Code Scanner button instead, the QR code scanner frame will appear. The system will check if user has available camera, the system will call the QR Code Scanner subroutine. If the user does not have available camera, the message box with “you don’t have a camera” text will popup.

The application will exit if user click window ‘X’ button.

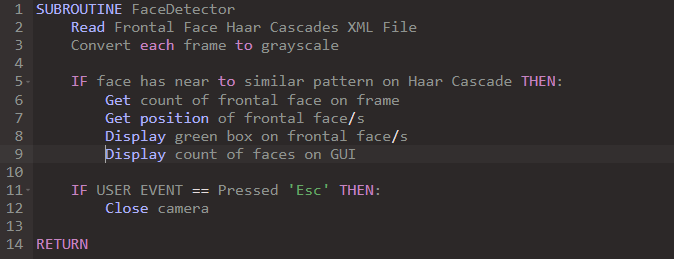
****

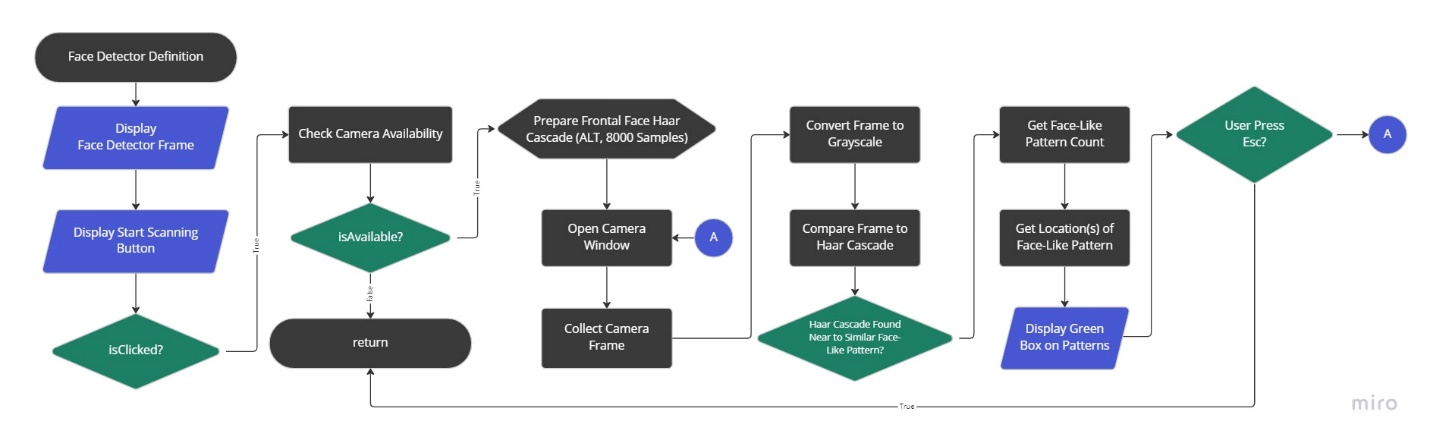
****

**Face Detection Data Frame**

This algorithm indicates the process flow of the face detection data frame. Initially, it reads the frontal face Haar Cascade’s XML file. Then, each frame being grabbed from the camera stream will be converted into grayscale for better accuracy. Then, if the face has near to similar pattern on Haar Cascade. After that, the logic will count and get the position of the detected frontal face on the frame. Then, the green box will also appear based on the position of faces. Lastly, it will display the count of faces on the GUI.

If user decided to click the ‘Esc’ button, the camera and current frame will exit. Then the user will be navigated to the main panel of the system again.

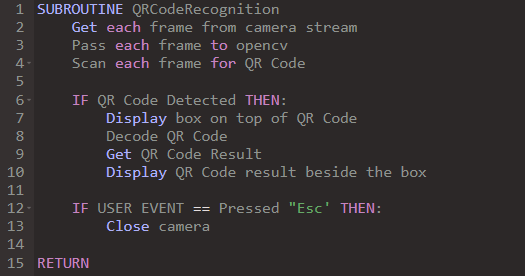


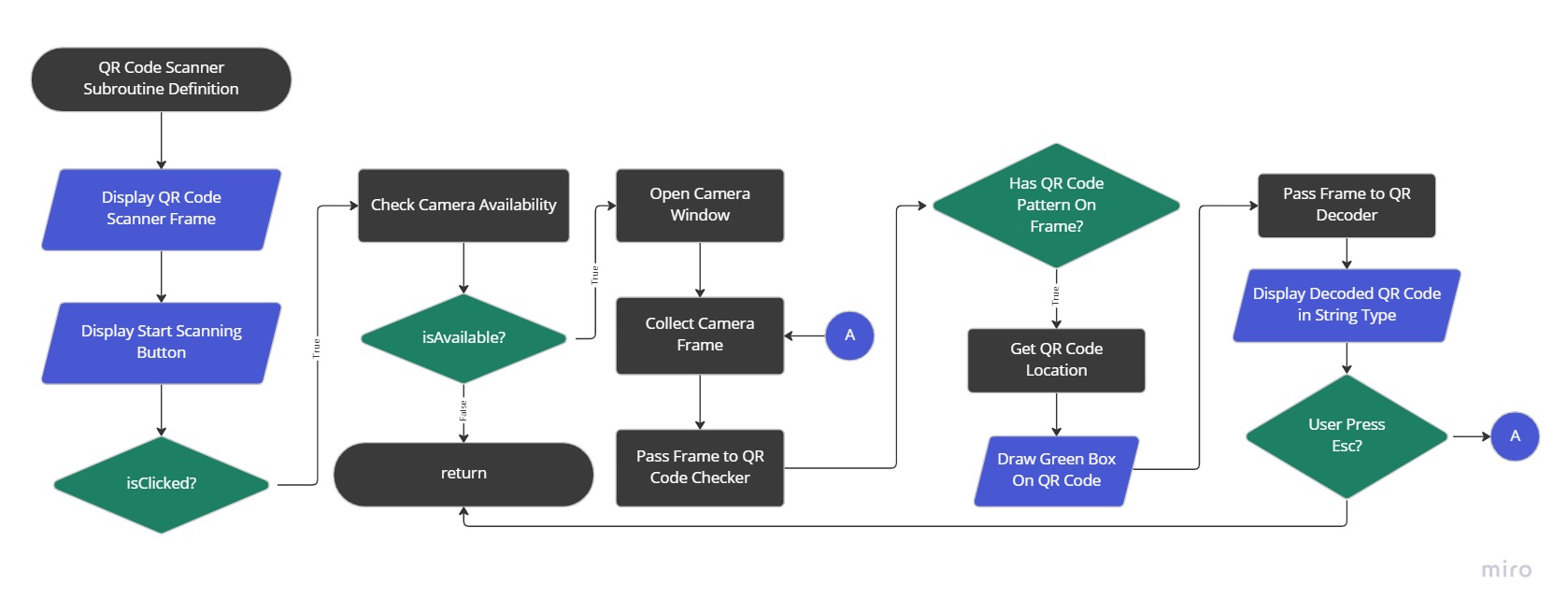


**QR Code Scanner Data Frame**

This algorithm shows the flow of the Quick Response scanner data frame. The logic behind this feature is basically grabs each frame from the camera stream and it is being passed to the open-source computer vision (open-cv) library and scan the frame if there are any QR code. If there is QR code, then the system will get the position of the QR code and display a box on top of it. Moreover, the open-cv will decode the QR code and get the string type result. Lastly, the system will display the result right beside the box.

If user pressed ‘Esc’, then it will close the camera and will go back to the main frame.





**Libraries**

**1. OpenCV -** a free and open-source computer vision library that provides tools for image and video processing**.**

**1.1 Frontal Face Haarcascade V2 –** a model of nine thousand face samples compiled in one XML file for object detection and face recognition functions.

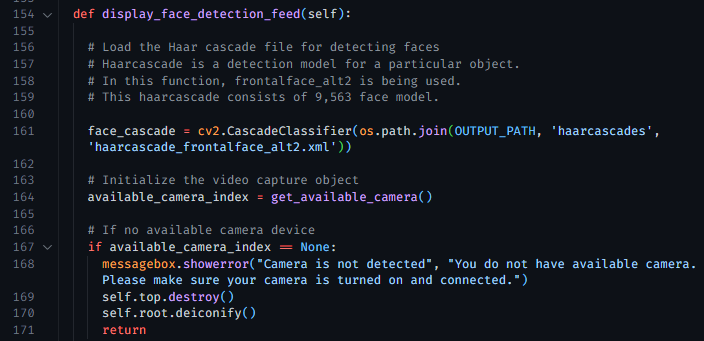
**2. TKinter -** is a Python library for building graphical user interfaces (GUIs) with the Tk toolkit. It provides tools for creating buttons, labels, text boxes, and menus, as well as support for layout management and event handling.

**Project Features Source Code (Back-end)**

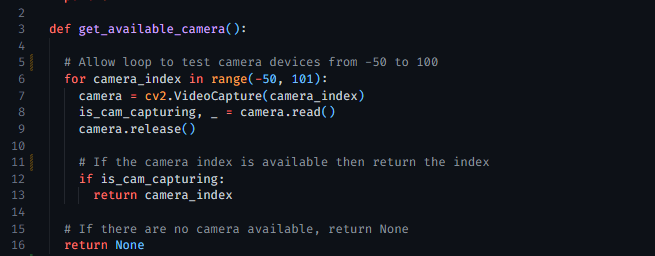
This portion discusses the back-end source code of the Face Detection and QR Code Scanner feature. The developers ensures that the source code is properly commented and the tried to manage to make it understandable to anyone.

**Face Detection Feature**

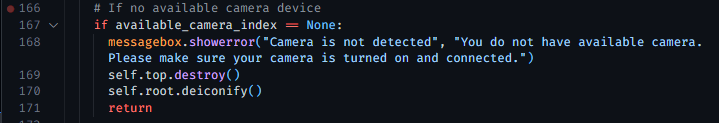
**Initialization of frontal face haarcascade second alternative**



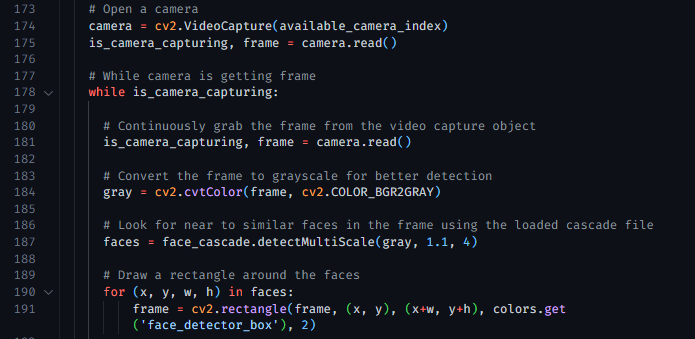
**Definition of get\_available\_camera()**



**On get\_available\_camera() return None**



**Face Detection under the hood process**

****

**Displaying frames on the camera stream**



**On user termination**





**Applying UI modification on the frame**

Text

Description automatically generated

**QR Code Scanner Feature**

Open a camera stream if user has available camera device

Text

Description automatically generated

Grab each frame from the camera

Graphical user interface, text

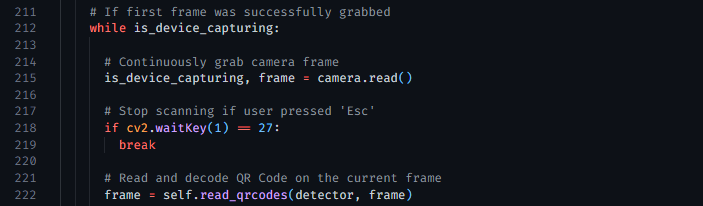
Description automatically generated

Initialize the OpenCV QR Code Detector

Graphical user interface, text

Description automatically generated

On camera continuous capture



Definition of user-defined read\_qrcodes function

\* It decodes frame and get the data text and box position  
 \* Send decoded text to GUI frame and update the GUI  
 \* If the frame detects no QR code, return the plain frame  
 \* Else, get the position and draw a rectangle and display text on the screen

Text

Description automatically generated

Displaying the returned frame from read\_qrcodes function on the video stream



On user pressed ‘Esc’ (refer to line 218), stop capturing and destroy windows



***\* END OF THE DOCUMENTATION \****

**Addendum**

**Decode Team as Developer**

The source-code of this project was developed by the Group Four (4) of the sophomore students’ section three (3) of the Bachelor of Science in Computer Science in Polytechnic University of the Philippines. This project is designed, coded, and developed duly for the academic purposes only. The team “Decode” managed to choose open-source assets and libraries in accordance with the best of our knowledge. Any critiques and suggestions are open for the public.

**Source Code released to the Public**

The team “Decode” allows anyone to clone and download the source-code of this project under certain conditions:

1. Anyone can modify, edit, and improve the content of the system.  
 2. We, “Decode” team, are not accountable to any damages (if this system can) to any hardware or software.

3. No one can sell and use this project for a commercial use.

Source Code Link: <https://github.com/markcalendario/face-detector-and-qrcode-scanner>