

# Facial Recognition Access Control System Project Report by

Hoda Ahmed;  
Student ID: C00214991

April 20, 2020

## **Abstract**

The purpose of this Final Project document is to provide a description of the final status of the project. The document will contain an introduction to the project, a detailed description of the submitted project, as defined by the specification and design documents. The document also contains the learning achievements and an overall review of the project.

# Contents

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introduction</b>                            | <b>4</b>  |
| <b>2</b> | <b>Project Description</b>                     | <b>4</b>  |
| 2.1      | Overview & Description . . . . .               | 4         |
| 2.2      | System Components . . . . .                    | 5         |
| 2.2.1    | Hardware Components . . . . .                  | 5         |
| 2.2.2    | Software Components . . . . .                  | 5         |
| 2.3      | System Architecture . . . . .                  | 6         |
| 2.4      | Desktop Application User Interface . . . . .   | 7         |
| 2.5      | Web Application User Interface . . . . .       | 21        |
| <b>3</b> | <b>Conformance to Specification and Design</b> | <b>24</b> |
| <b>4</b> | <b>Learning Outcomes</b>                       | <b>25</b> |
| 4.1      | Technical Achievements . . . . .               | 25        |
| 4.1.1    | Electronic Circuits . . . . .                  | 25        |
| 4.1.2    | Python . . . . .                               | 25        |
| 4.1.3    | Flask, Jinja2 & Bootstrap . . . . .            | 25        |
| 4.1.4    | Raspberry Pi . . . . .                         | 25        |
| 4.1.5    | PyQt . . . . .                                 | 26        |
| <b>5</b> | <b>Project Review</b>                          | <b>27</b> |
| 5.1      | Positive Aspects/Aspects Achieved . . . . .    | 27        |
| 5.2      | Aspects Not Achieved . . . . .                 | 27        |
| 5.3      | Aspects Gone Wrong . . . . .                   | 27        |
| 5.4      | Things I Would Change . . . . .                | 27        |
| 5.5      | Problems Encountered . . . . .                 | 28        |
| <b>6</b> | <b>Future Features</b>                         | <b>29</b> |
| <b>7</b> | <b>Acknowledgements</b>                        | <b>29</b> |

# 1 Introduction

The following document will provide a detailed description of the overall progress and final status of the Facial Recognition Access Control project undertaken by the author.

The project description will cover a detailed outline of the features and functionalities that the project aimed to satisfy, including hardware, software and user interfaces. Description of conformance to specification will discuss the final outcome of the project with regards to the initial proof of concept and anticipated design of the project.

## 2 Project Description

### 2.1 Overview & Description

The facial recognition access control system is a system that uses multi-factor authentication, including biometric authentication, to grant users access to equipment rooms. The system will be used to control unauthorized access to equipment rooms in corporate buildings, through a Raspberry Pi computer.

This system will use Amazon AWS Rekognition API to process users' faces and determine whether they should be granted access to the rooms or not.

The system will also keep a log of the users that have been granted access to the rooms and the date/time at which the room was accessed. Denied access to users will also be logged. Administrators will be able to view these logs and view user information and other related information through the web application that will be created. The web application may also be used by normal users to update their information and view their own access logs.

The system is centred around the Raspberry Pi computer, which will carry out most of the functionalities of the system. The Raspberry Pi has a desktop application installed on it, through which users can login and register to access the room. The Raspberry Pi also controls the lock, by providing it with power to pull back/release its tongue, hence unlocking the door and granting access to users.

A Flask webserver also runs on the Raspberry Pi, and it allows administrators to log into it and view the access logs that have been attempted since the launch of the system. This also includes failed login attempts.

As explained thoroughly in the Research Document for the Facial Recognition Access Control System, there are a few solutions that already exist in the market that offer the same service offered by this project. However, the unique aspect about this FRACS project is that it offers access control, taking into consideration both security and privacy. The other solutions that exist in the wild integrate facial recognition into their full monitoring suite, storing captured images and videos, and not fully considering the privacy of the system users. This is not the case for FRACS, as the only information stored on them is a face ID, and the image captured is stored only for the duration of the processing and is permanently deleted after. And this is a great advantage to us.

## 2.2 System Components

### 2.2.1 Hardware Components

The hardware components of the FRACS project have been examined in detail in the Functional Specification document, section 2.3.1. These components that are used in the FRACS project are the following:

- **Raspberry Pi:** The Raspberry Pi is a small-sized computer that is mainly used for learning computer-related material. It is the central component of the whole project, where the desktop application is installed. It is designed for users to enter their credentials to verify themselves. The Raspberry Pi will also be responsible for providing power to the physical lock that will be attached to it.  
The Raspberry Pi has a GPIO board that's outline below, with the purpose of each pin in the described in the image.
- **Raspberry Pi camera:** The camera provides the capability of taking pictures. It will be attached to the Raspberry Pi to capture the user's face trying to access the equipment room.
- **LCD touchscreen:** This touchscreen will provide the user interface and will be the interactive point for the user.
- **Solenoid lock:** The solenoid lock will be responsible for locking/unlocking the door of the equipment room. It will be attached to and controlled by the Raspberry Pi, by receiving power from it and unlocking the door when given a signal and when the door is expected to unlock.
- **Relevant cables and connectors:** e.g. for connecting Raspberry Pi to LCD touchscreen, Raspberry Pi to solenoid lock, and solenoid lock to door.

### 2.2.2 Software Components

- **Rekognition API:** API to be used for all the facial recognition operations e.g. face detection, normalization, feature extraction etc.
- **Twilio SMS API:** As mentioned previously, the FRACS project harnesses the Twilio PaaS to send SMS messages when an admin is attempting to register a new user. When an account is registered to Twilio, it receives an account SID and an authentication token. Using these credentials and when the Raspberry Pi is triggered, it sends an SMS code via HTTP to Twilio's servers, who forward the message in SMS format to the admin's phone.
- **Flask Server:** Flask is a microframework, used for building web applications. This framework was considered the best option for development on this project, as it is extremely lightweight, and is specifically for Python, which is a core development language of the Raspberry Pi. This Flask webserver runs on the Raspberry Pi, and it presents administrators with the access logs for FRACS.

## 2.3 System Architecture

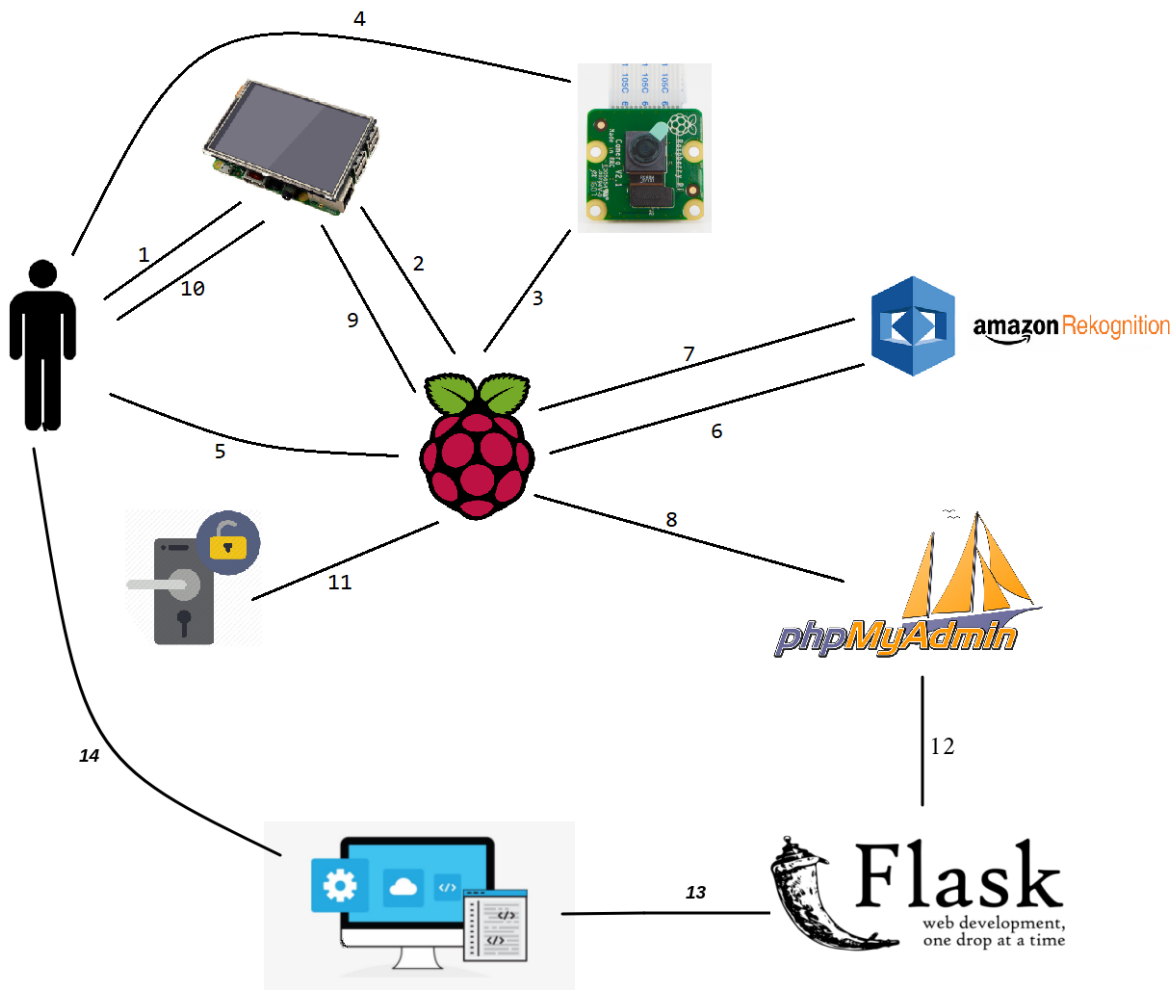


Figure 1: FRACS System Architecture

The facial recognition access control system is centered around the Raspberry Pi which is situated in a strategic position in the system. Most of the interactions of the system are carried out with/on/using the Raspberry Pi.

As outlined in the diagram above, the user interacts with the LCD touchscreen to trigger the starting of the desktop application on the Raspberry Pi and logs in by entering their credentials. Following this, the camera is activated, and it takes a picture of the user. That picture is then stored on the Raspberry Pi which carries out the API calls to AWS Rekognition, sending the picture as a HTTP request. Once the API received the HTTP request, it detects the face in the image, extracts the facial features and stores it in a vector. That vector is compared against the existing vectors in the database to check whether a user exists (and is authorized to enter) or not. The result is returned to Raspberry Pi as a HTTP response, and it contains a face ID of the person. Once it receives the face ID, the Raspberry Pi checks for it in the local database along with the credentials entered at the start by the user. If the face ID/credentials combination exists, the Raspberry Pi sends a signal to the lock, triggering it, and unlocks the room for the user. The Raspberry Pi also records a new access entry in the logs database. However, if the face ID/credentials combination doesn't exist, then the user is denied access and a failed access attempt is also logged.

## 2.4 Desktop Application User Interface

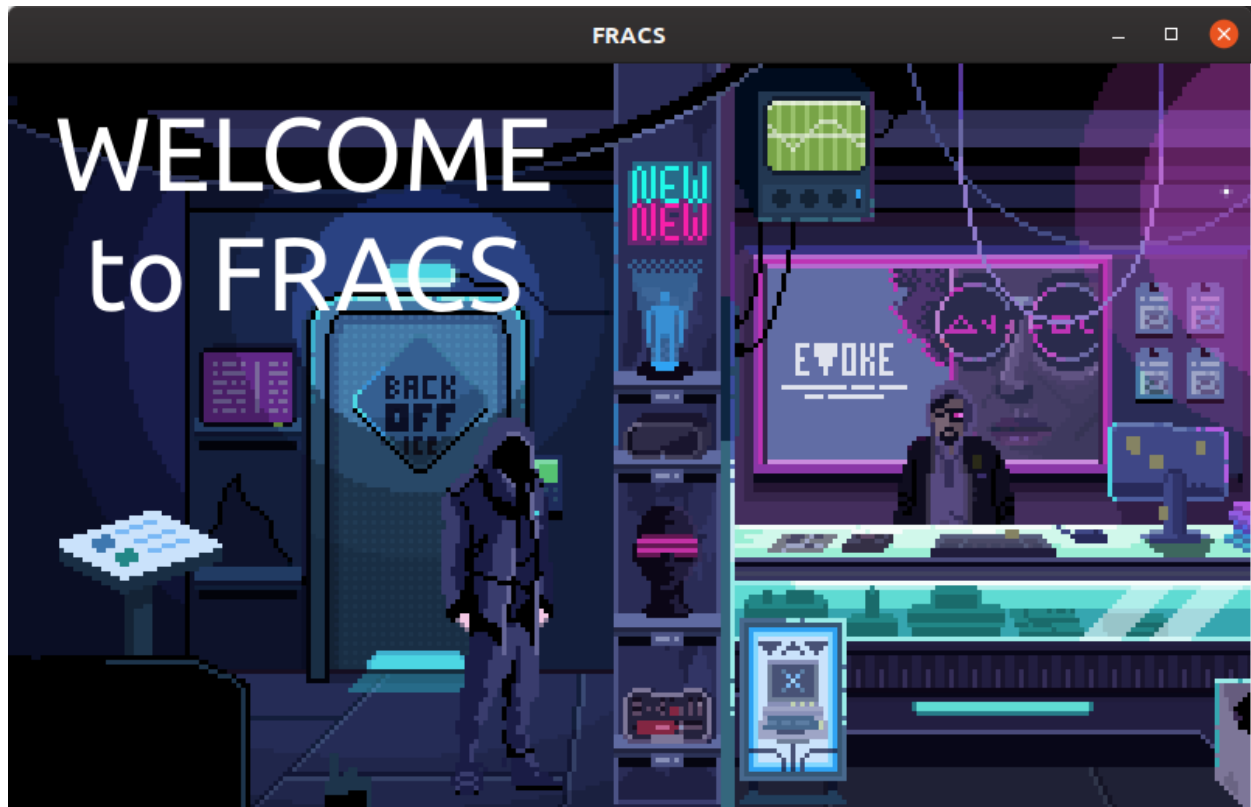


Figure 2: FRACS Desktop App - Home

The home page is designed with a catchy, yet calm background. It is a splash screen that appears for only 3 seconds, before displaying the main menu, where users can choose whether to login or register.

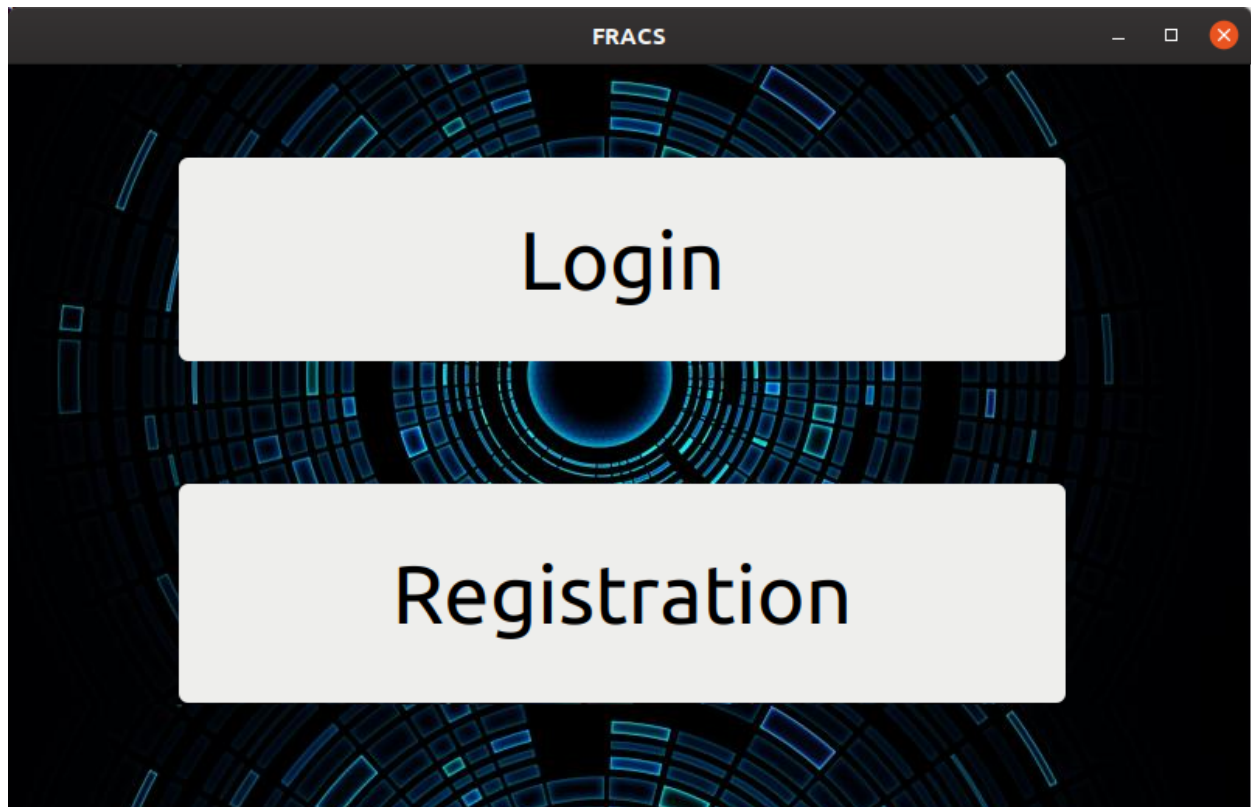


Figure 3: FRACS Desktop App - Main Menu



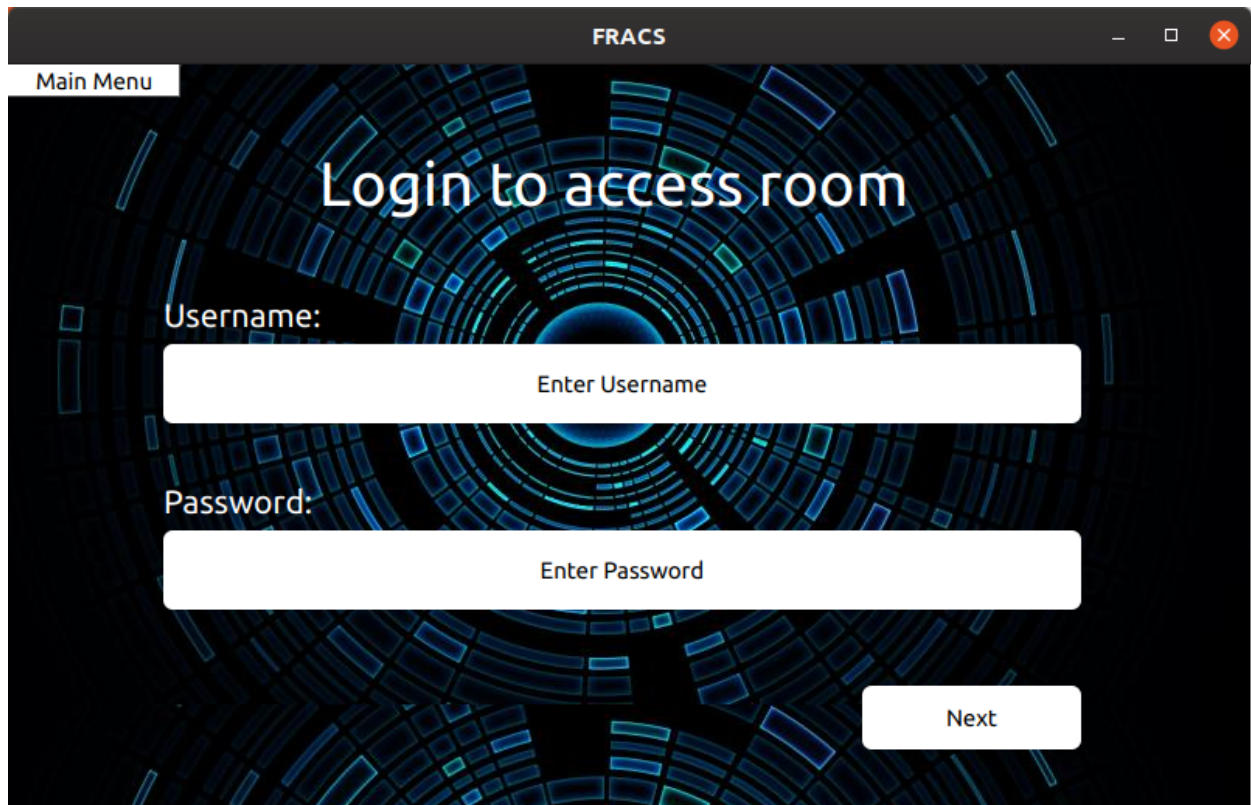


Figure 4: FRACS Desktop App - Login

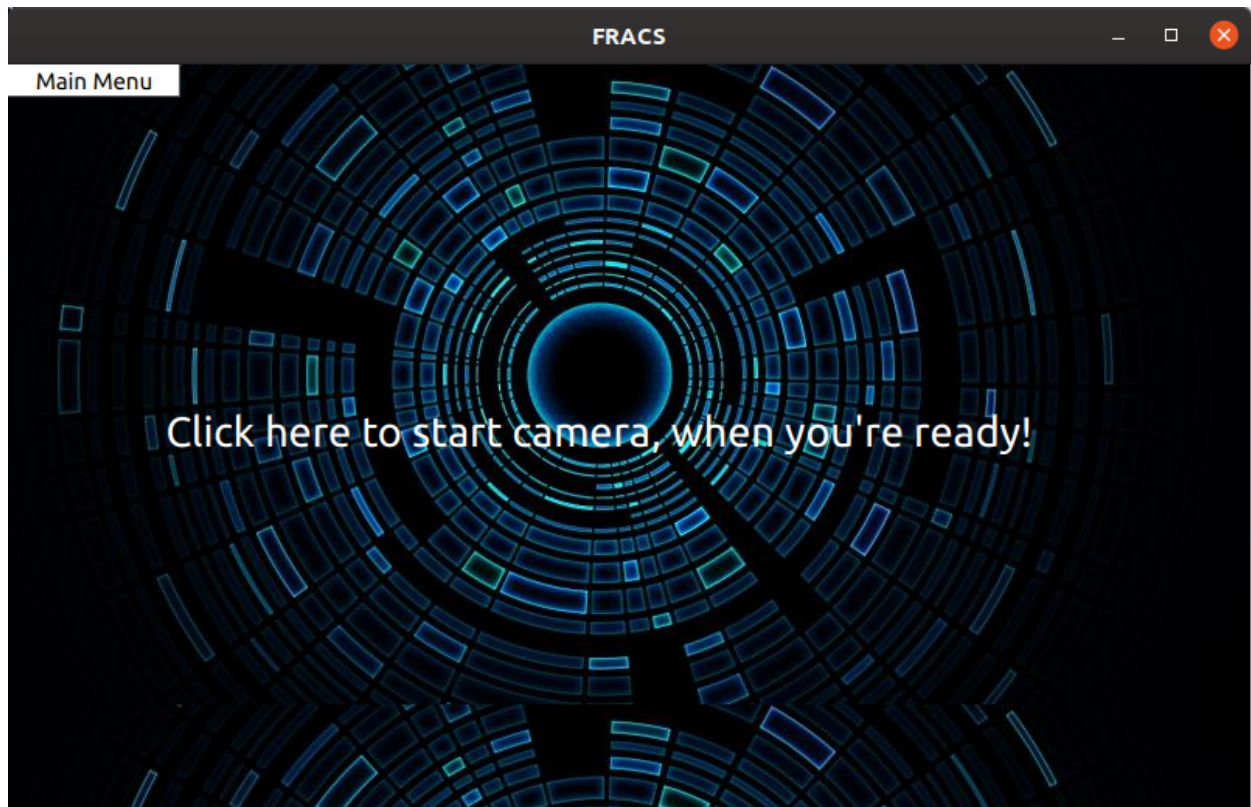


Figure 5: FRACS Desktop App - Camera Page

After clicking on this screen/tapping on it, a camera preview is displayed to the user, along with a countdown of 3. After the countdown is over, the camera preview captures an image and carried out the respective functionalities.



Figure 6: FRACS Desktop App - Login Success

The above screen will display to the user in the event that their login attempt is successful. This is shown when the login credentials and the face ID match.

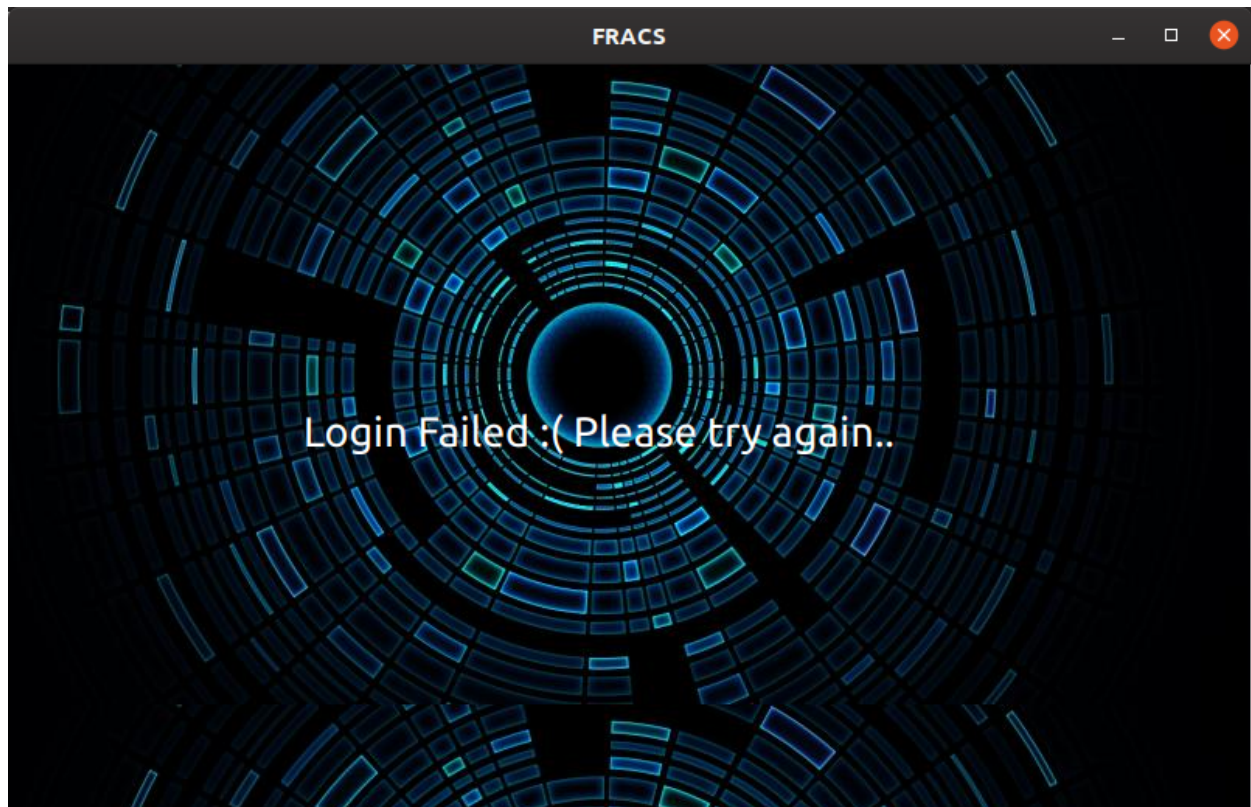


Figure 7: FRACS Desktop App - Login Fail

The above screen will display to the user in the event that their login attempt is NOT successful. This is shown when the login credentials and the face ID DON'T match.



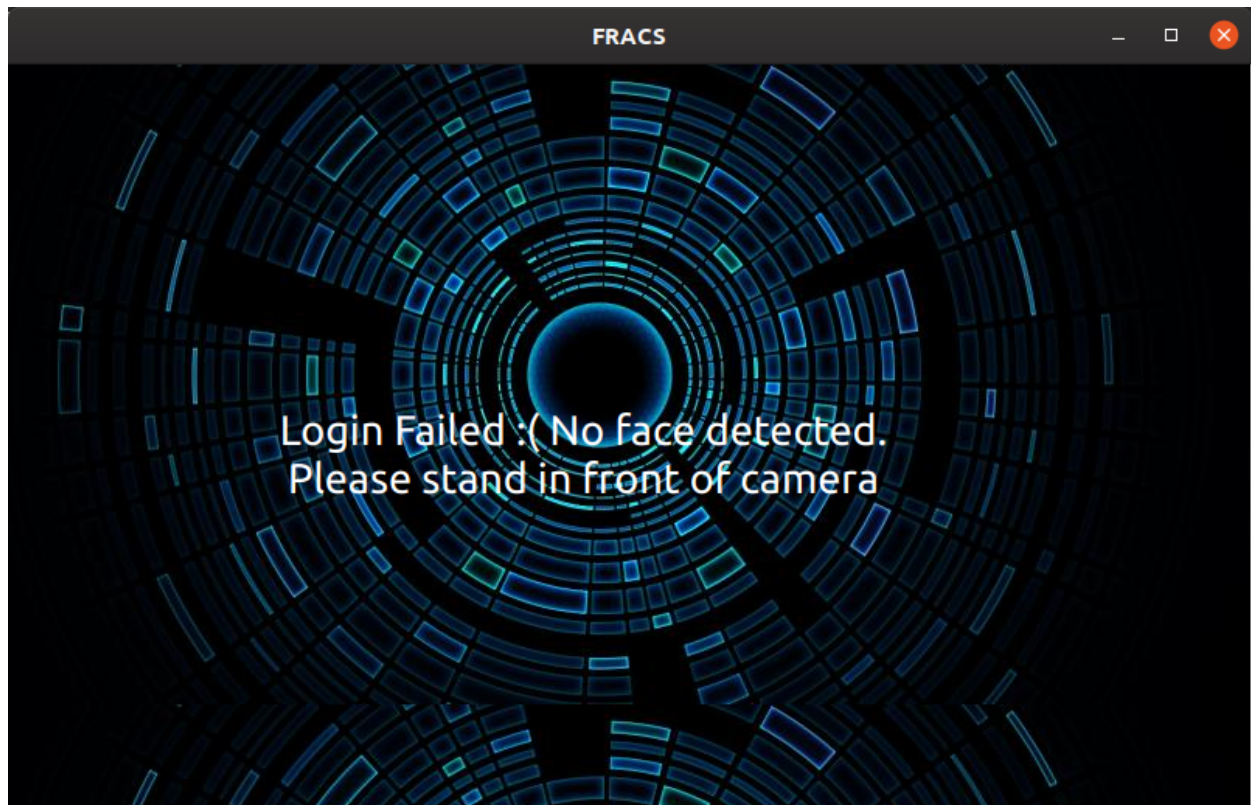


Figure 8: FRACS Desktop App - Login Fail 2

The above screen will display to the user in the event that the camera cannot detect any faces in the image captured.

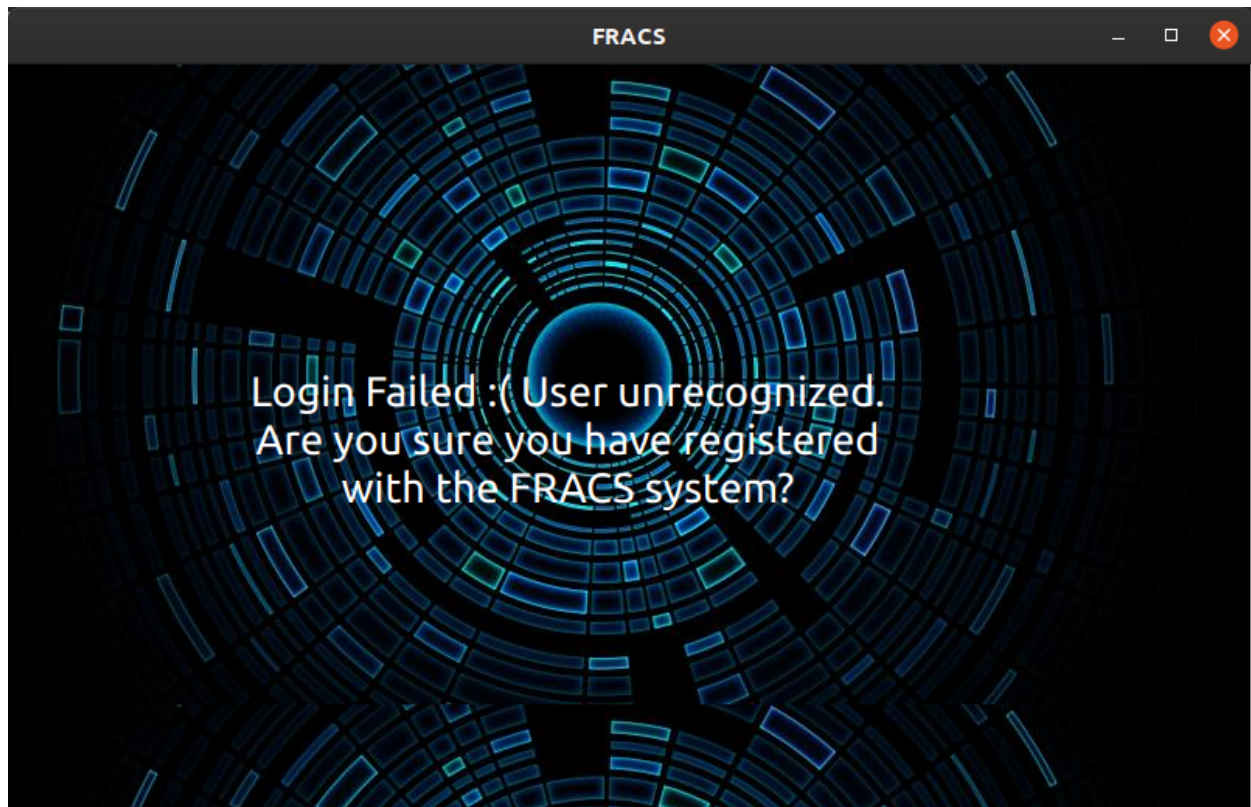


Figure 9: FRACS Desktop App - Login Fail 3

The above screen will display to the user in the event that the camera cannot identify the user face detected in the image captured.

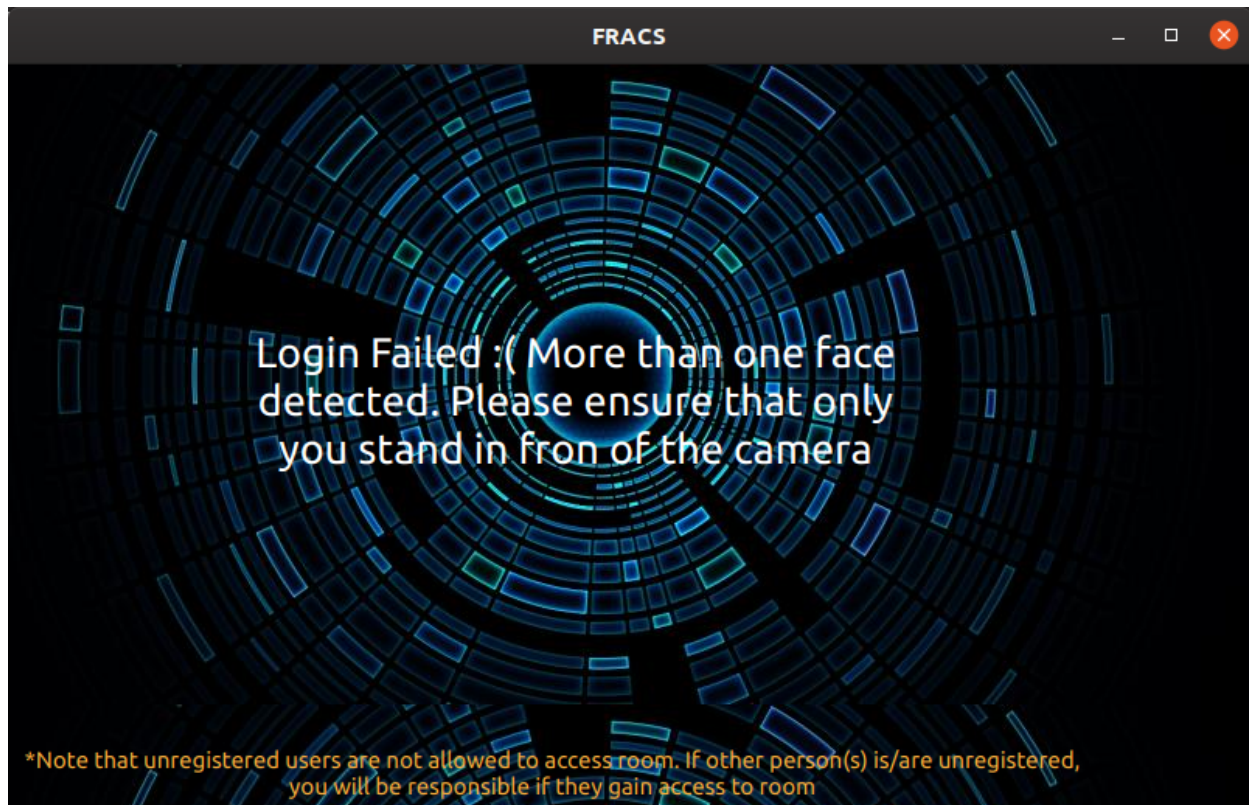


Figure 10: FRACS Desktop App - Login Fail 4

The above screen will display to the user in the event that the camera detects more than one face in the image captured.

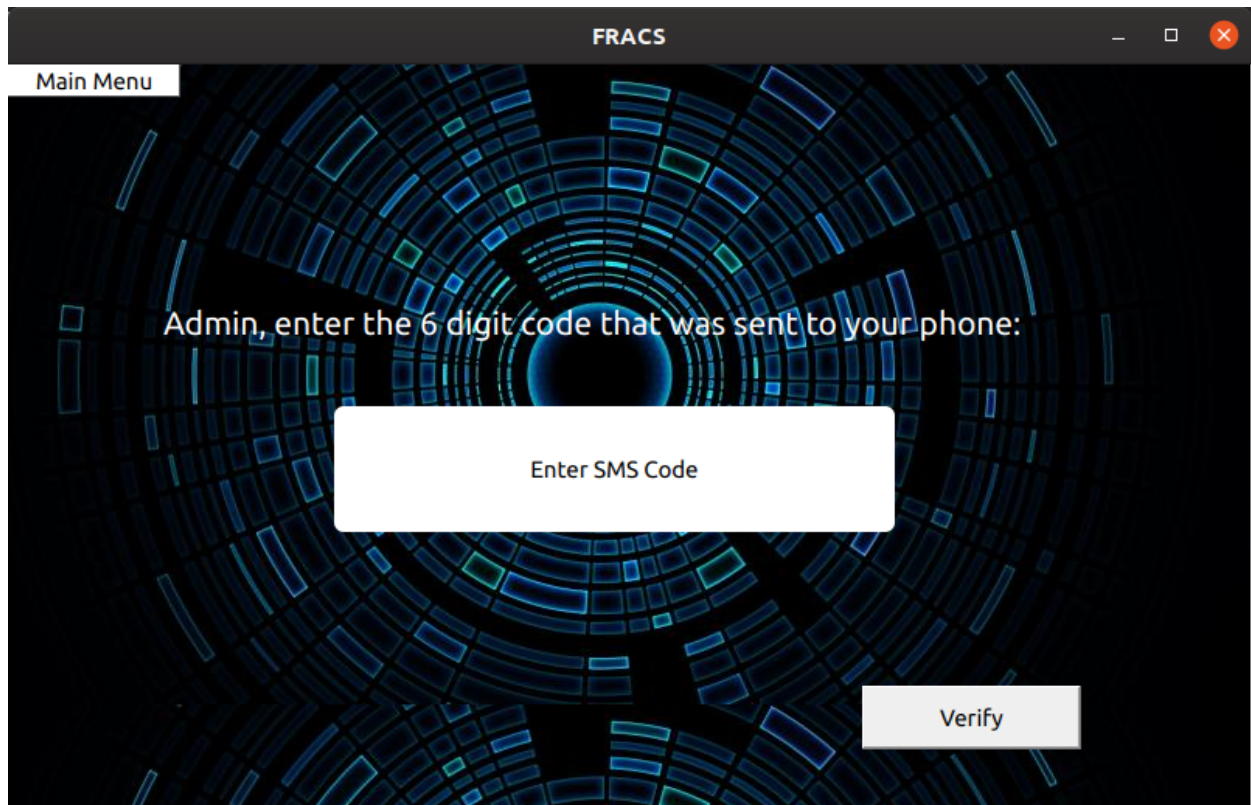


Figure 11: FRACS Desktop App - SMS Code

Admin enters the SMS code they received to their phone here to verify themselves when they wish to register a user.



FRACS

Main Menu

# Register to FRACS

First Name:

Enter First Name

Last Name:

Enter Last Name

Login Username:

Enter Username

Login Password:

Enter Password

☐ Admin?

Next

Figure 12: FRACS Desktop App - Registration

New user enters their information here to be stored to the local database, and then the camera is started so that they can get their face scanned and detected.

FRACS

Main Menu

Please enter your phone number in the below field, then click 'Send Code'.  
You will get a 6-digit code to that phone number, which you will need to enter below.

Enter Phone Number in this format: +353xxxxxxxxx

Send Code

Enter SMS Code

Verify + finish!

Don't have your phone? You can skip this part and do it on the FRACS Website later.  
However, you will not be able to register new users until you verify your phone number!

Skip

Figure 13: FRACS Desktop App - New User SMS Code

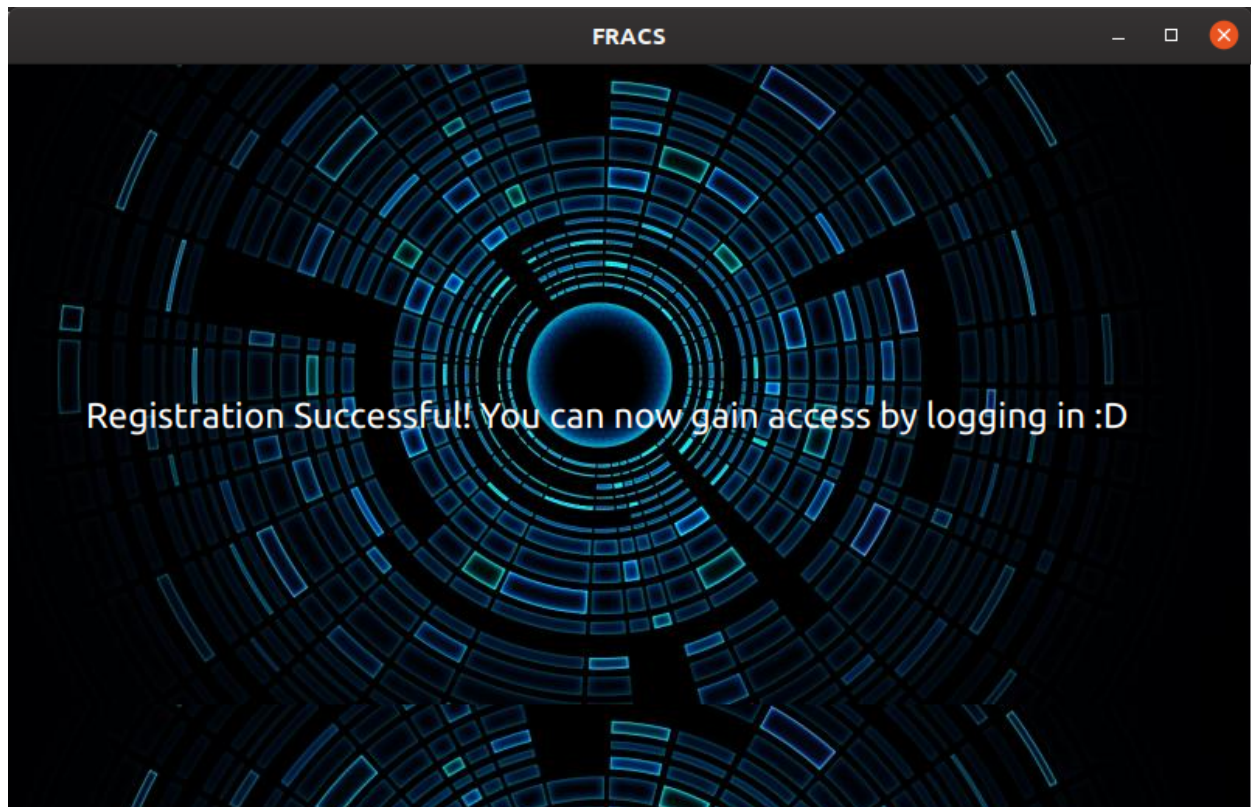


Figure 14: FRACS Desktop App - Registration Success

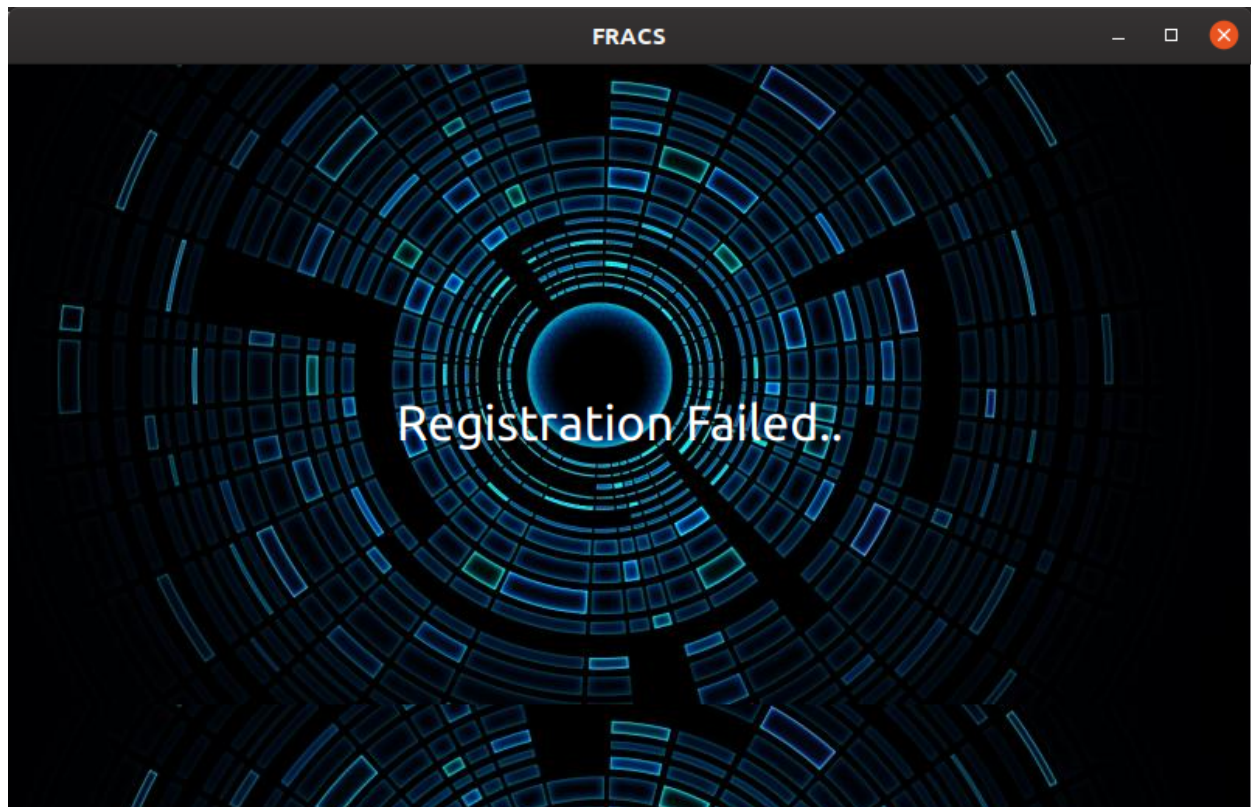
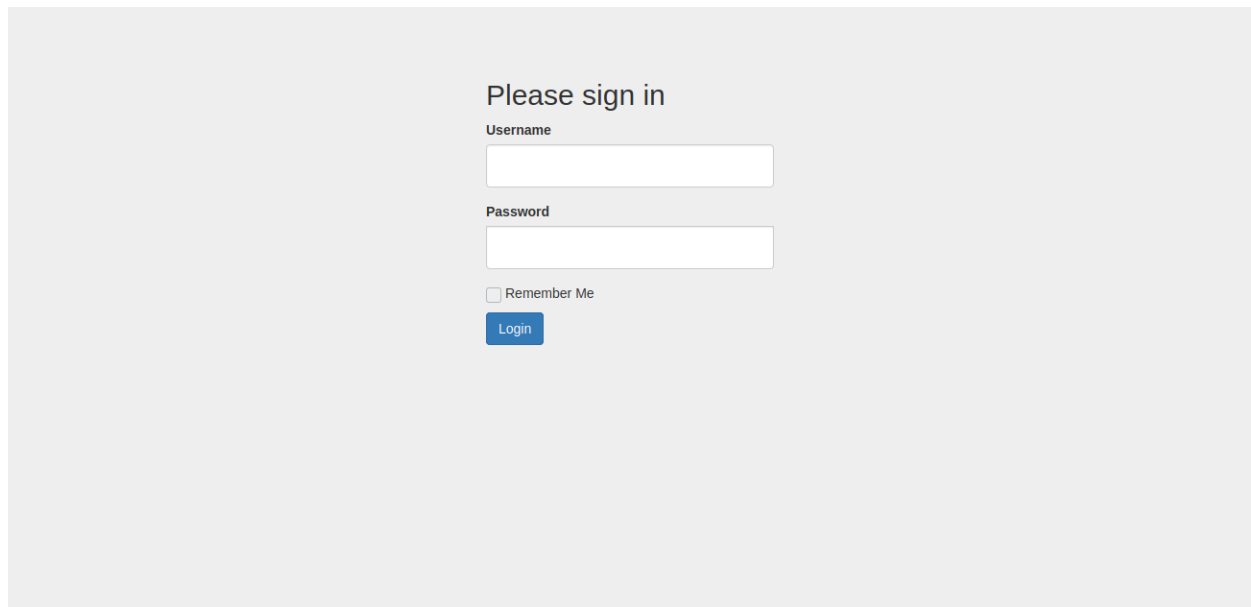


Figure 15: FRACS Desktop App - Registration Fail

## 2.5 Web Application User Interface

### Login Page



Please sign in

Username

Password

☐ Remember Me

Login

Figure 16: FRACS Web App - Login

### Dashboard

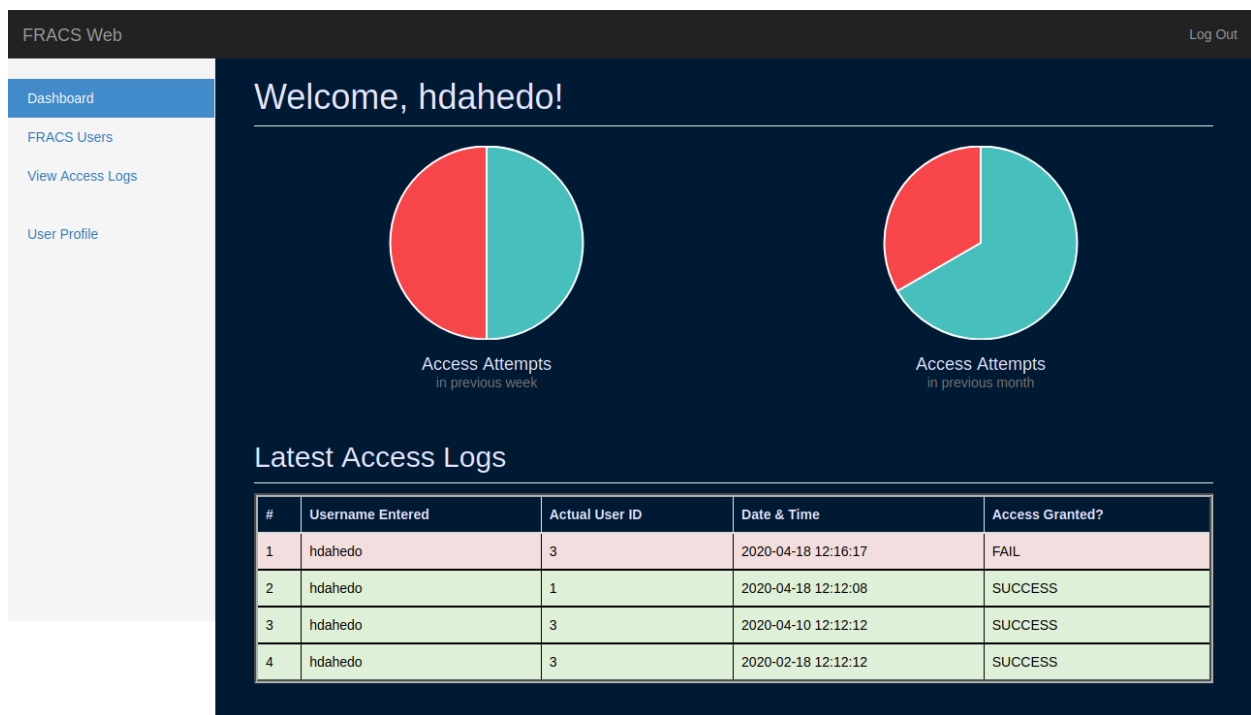


Figure 17: FRACS Web App - Dashboard

## Users List

FRACS-Web Log Out

Dashboard

FRACS Users

View Access Logs

User Profile

### FRACS Users

Show 10 entries Search

| ID | Name        | Username | Admin? | Phone Number  | Face ID            | Admin Registered |
|----|-------------|----------|--------|---------------|--------------------|------------------|
| 1  | hoda ahmed  | hdahedo  | True   | +353899791573 | uy838vksjdhiuq3DgC | None             |
| 2  | johnny depp | jdepp    | False  | None          | AEJU6FKBVFUJNVGUJN | hoda ahmed       |

Showing 1 to 2 of 2 entries Previous 1 Next

Figure 18: FRACS Web App - Users List

## Logs List

FRACS Web Log Out

Dashboard

FRACS Users

View Access Logs

User Profile

### Access Logs

Show 10 entries Search

| # | Username Entered | Actual User ID | Date & Time         | Access Granted? |
|---|------------------|----------------|---------------------|-----------------|
| 1 | hdahedo          | 1              | 2020-04-18 12:12:08 | SUCCESS         |
| 3 | hdahedo          | 3              | 2020-04-18 12:16:17 | FAIL            |
| 4 | hdahedo          | 3              | 2020-02-18 12:12:12 | SUCCESS         |
| 5 | hdahedo          | 3              | 2020-04-10 12:12:12 | SUCCESS         |

Showing 1 to 4 of 4 entries Previous 1 Next

Figure 19: FRACS Web App - Logs List

Current User Profile

FRACS Web

Log Out

Dashboard

FRACS Users

View Access Logs

User Profile

User Profile

User ID

1

Login Username

hdahedo

Face ID

uy838vksjdhiuq3DgC

First Name

hoda

Last Name

ahmed

About You

An admin/user

Figure 20: FRACS Web App - User Profile

### 3 Conformance to Specification and Design

In this segment of the document, I will contrast the submitted proposal and specifications with the project as it is today.

Overall, the final project submission conforms almost precisely to what was intended in the initial proposal and subsequent final specification. All of the core functionality is provided in the desktop application.

Initially, at the start of development and during one of the discussions with my supervisor, James Egan, we thought of including a motion sensor in the project, that would automatically start the desktop application on the Raspberry Pi, once a user appears in front of the camera. After further discussions and meetings, we came to the conclusion that this "feature" was a little unnecessary. This is for a number of reasons: 1. depending on where the camera is placed, there will be movement around, hence, triggering the motion sensor will be a continuous action; and 2. it is best to let the user interact with the system as much as possible, so that they can login and register.



## 4 Learning Outcomes

I have to admit that there were a lot of learning outcomes that resulted from researching and working on this project for more than 5 months. From the very beginning, I wanted to push my boundaries and expand my horizons in terms of technologies I am familiar with. Having nearly zero experience with Python prior to working on this project was a huge factor that led to a delay in project development.

Electronic circuits were also extremely challenging. I must mention that the burnt LCD touch-screen and old Raspberry Pi, as a result of a poorly set up electronic circuit, lead me to understand how complex and dangerous these circuits are.

However, I believed that this was the best chance to pick up on new technologies, as support and guidance from lecturers was sought and was generously provided.

### 4.1 Technical Achievements

As previously stated, I have nearly zero experience in Python, electronic circuits and working with Raspberry Pi's. However, learning some technologies and becoming familiar with them were harder than other technologies and had a steeper learning curve.

#### 4.1.1 Electronic Circuits

There were a few sources for me that helped with learning electronic circuits. The most memorable and convenient source was an Electronics Engineering lecturer, who teaches at Institute of Technology Carlow. I met twice with the said lecturer, where he explained to me how basic circuits work and as a way to get started, assisted me in getting a simple LED to flash using the Raspberry Pi. I later took this example and altered it, replacing the LED with the solenoid lock and attaching a battery to see if it would work.

#### 4.1.2 Python

Python was not too difficult to learn and definitely didn't have a learning curve as steep as electronic circuits. To learn Python, I took an online course on CodeCademy.com:

<https://www.codecademy.com/learn/learn-python>

While time did not allow me to complete this course, the time that was invested into it gave me the confidence to move forward with my project, and start developing right away.

#### 4.1.3 Flask, Jinja2 & Bootstrap

Using Flask gave me insights on how easy web development can become if using the right frameworks. Jinja2, the Python template engine used to create HTML, amazed me because of how simple it made the backend-frontend transition proceed smoothly. Like electronic circuits, Flask took a long time to understand and personally, out of all the new technologies I came across, it had the steepest learning curve.

#### 4.1.4 Raspberry Pi

Although it is very different from a Raspberry Pi, I have little experience with Arduino boards. This noted, it did help me to a certain extent in understanding how a Raspberry Pi works, especially it's GPIO board (general purpose input/output board).

#### **4.1.5 PyQt**

With only a very small amount of background in desktop application development using Java AWT & Swing, PyQt was very challenging to grasp. I am grateful that Qt Designer exists, as it was a very important source for learning PyQt development, and not just to develop.

## 5 Project Review

This segment of the document will review aspects of the project that went positively and negatively.

### 5.1 Positive Aspects/Aspects Achieved

Personally, I would consider the entire project a success. All the core functionalities that were defined at the start are completed, except for a few minor parts. Opposite to what I thought, assembling the lock and attaching it to the Raspberry Pi and the battery took less than a day, and it was, unexpectedly, successful (wait for a small surprise in the Aspects Gone Wrong section of this document).

Midway during development I thought that I would not be able to finish the Desktop application thoroughly because of how slow my progress was and how much time was invested in other college projects. However, I was proven wrong, and the only part of the project that was completed to the full was the desktop application.

### 5.2 Aspects Not Achieved

Designing a FRACS mobile application was on the bucket list for this project, but being wary of the limited time and the amount of information I will need to absorb to use the new technologies, I predicted that I will not be able to create the said mobile app.

### 5.3 Aspects Gone Wrong

While testing the functionality of the solenoid lock for the project, I accidentally pulled out the ground wire, which lead to a (disappointing) burning of the LCD touchscreen and the Raspberry Pi. Fortunately, I had a spare Raspberry Pi so it replaced the burnt one, and I had to buy a new LCD touchscreen. Nevertheless, the lesson was taught - properly secure all wires, and **ALWAYS**, remove the ground wire last, after plugging out everything else from the Raspberry Pi4.

### 5.4 Things I Would Change

If I had the chance to start working on this project from the start, there are a few things I would change to make it work better.

- **Time management & development inconsistency:** The main problem I faced while working on the FRACS project is time management. At the start there was a lack of consistency from the development side. I would begin a task and after a short while put it aside and begin another task, without finishing the first task. What made it worse was the fact that I interchangeably worked on different tasks using different technologies e.g. first task was the GUI for desktop application, while second task was backend for web application. // I would change this, and be sure to finish off one task first before moving to the other.
- **Early start:** I have started development mid-December 2019. This may seem like enough time to finish the project on time, however, I should have kept in mind other deadlines and

submissions for other modules in college. Due to working on different projects and assignments at the same time, progress on the FRACS project was extremely slow. Eventually, mid-March 2020, I was able to get back on track again and follow the project plan set out in the Design Manual document.

If I could restart my project again, I would start development a month earlier, to give myself extra time to think about other features that I can include in the project, as well as add new valuable features.

- **Do more research on Flask & Web Development:** Web development is one of the things I really dislike working on, but using Flask made me realize that it isn't as bad as I thought. The plan I had set out to learn more about Flask and how to use it specified a short period of time. I believe that if I had spent more time learning Flask, I would have produced a more enhanced version of the web application.

## 5.5 Problems Encountered

As expected, there were a lot of problems encountered during the development of the FRACS project.

I decided to start development of the GUI on laptop first, to get a feel of what PyQt development is like. However, for some unknown reason, MySQL would not work on the laptop and would display an error. To get around this problem, I decided to use a Linux virtual machine instead to get started. Not surprised that another issue had appeared, there was a problem with the VM accessing the laptop camera, so I had to use a static image instead of using the camera in realtime. This was a problem as I couldn't work with the application like how it would work natively on the Raspberry Pi.

## 6 Future Features

As explained above, there was a plan to make a mobile application for admins, which was going to be an easier and more convenient method of viewing access logs and user information. As mentioned above, also, this was not a confirmed plan - I was convinced that time will not suffice to produce a mobile application.

Therefore, for the future features, I am planning to introduce a mobile application to make life easier for admins.

There was also another future feature that was discussed with my supervisor, James, and it was to add another layer of security to the system, by using another kind of biometric authentication. I loved this idea and suggested adding the finger vein biometric authentication layer (see Research Manual document). However, this would only be added to the project after the mobile application is implemented.

## 7 Acknowledgements

There are a lot of people that I have to acknowledge in this section to show my gratitude and respect. The support and encouragement that I have been receiving from the Cyber2020 class is something I will never be able to forget. The genuine assistance and reassurance that we'll all be just fine was valuable for my survival through this year.

Furthermore, I would like to express my appreciation to my close friend, Aya Aloraimi, who has helped me keep going since first year in college and remained supportive throughout the college years.

Also, I would like to say a huge thank you to the Electronic Mechanics & Aerospace Engineering lecturer that provided me with the helpful info for operating the solenoid lock, Vincent O'Brien. Thanks to Vincent I had the relief of not burning one more Raspberry Pi computer and an LCD touchscreen!

All of the lecturers from first year to this year have my respect, and I would like to thank you all for sharing your enthusiasm and knowledge with us.

Finally, I want to take the time to thank my project supervisor, James Egan, for his advice, guidance, opinion, and support. He broadened the boundaries of my vision and capabilities for the duration of the project and was always very prompt. Thank you for the time and effort you put in for this project to be completed.