

Facial Recognition Access Control System Technical Manual by

Hoda Ahmed;
Student ID: C00214991

April 19, 2020

1 Introduction

The technical manual document contains all code for the FRACS application. The FRACS application can be run on any Linux/Unix-based system, if the suitable libraries are installed.

For the desktop application, all of the files are Python files. For ease of readability and structure, the files are broken down to represent database operations and external libraries and other APIs operations.

2 Desktop Application (Raspberry Pi)

2.1 Initialization

This section of the document includes code that is used to set up the database tables and create the collection in AWS. Firstly, the constructor is called, which calls the main function, invoking the following methods.

```
1 def main():
2     createDB()
3     createTables()
4     collection_id = "FRACSUsers"
5     createRekCollection(collection_id) # create collection called FRACSUsers
6     super_admin_face_id = indexSuperAdminFace(collection_id) # index super admin
7     # face & store it in FRACSUsers collection
8     createAdminDBEntry(super_admin_face_id)
9
10 if __name__ == '__main__':
11     main()
```

Listing 1: fracs-init.py - main()

```
1 #firstly, local database needs to be set up:
2 def createDB():
3     db_connection = mysql.connector.connect(host = "localhost", user = "hoda",
4     password = "XXXX")
5     db_cursor = db_connection.cursor()
6     db_cursor.execute("CREATE DATABASE IF NOT EXISTS FracsDB;")
7     db_connection.close()
```

Listing 2: fracs-init.py - Local DB Initialization

```
1 #then, tables initialized
2 def createTables():
3     db_connection = mysql.connector.connect(host = "localhost", user = "hoda",
4     password = "XXXX", database = "FracDB")
5     db_cursor = db_connection.cursor()
6     db_cursor.execute("CREATE TABLE 'fracs_users' ( "
7         " 'user_id' int NOT NULL AUTO_INCREMENT,"+
8         " 'firstname' varchar(255) NOT NULL,"+
9         " 'lastname' varchar(255) NOT NULL,"+
10        " 'login_username' varchar(255) NOT NULL,"+
11        " 'login_password' varchar(255) NOT NULL,"+
12        " 'is_admin' tinyint(1) NOT NULL DEFAULT '0'," +
13        " 'phone_number' varchar(15) DEFAULT NULL,"+
14        " 'face_id' varchar(255) NOT NULL,"+
15        " 'admin_registered' varchar(255) DEFAULT NULL,"+
16        " PRIMARY KEY ('user_id'))")
17
18    db_cursor.execute("CREATE TABLE 'fracs_logs' ( " +
19        " 'log_id' int NOT NULL AUTO_INCREMENT,"+
20        " 'username_entered' varchar(255) NOT NULL,"+
21        " 'face_id' varchar(255) NOT NULL,"+
22        " 'attempted_user_id' int DEFAULT NULL,"+
23        " 'login_datetime' datetime NOT NULL,"+
24        " 'is_successful' varchar(10) NOT NULL,"+
25        " PRIMARY KEY ('log_id'))")
26
27    db_connection.commit()
28    db_connection.close()
```

Listing 3: fracs-init.py - Local DB Tables Initialization

Following the initialization and creation of the local database, the Rekognition API will be called to create a collection for the users' faces, and then a "super admin" will be registered on the collection using the `index_face` function.

```
1 def createRekCollection(collection_id):
2     #Create a collection
3     print('Creating collection:' + collection_id)
4     response = client.create_collection(CollectionId = collection_id)
5
6     print('Status code: ' + str(response['StatusCode']))
7     if (str(response['StatusCode']) == "200"):
8         print("Collection created successfully!")
9
10    # as soon as a collection is created, an admin must register their face to be
11    # able to login & register other users
12    # therefore, with the creation of the collection, the admin must also store
13    # their face in that collection i.e. call indexSuperAdminFace method
```

Listing 4: fracs-init.py - Rekognition Collection Initialization

```
1 def indexSuperAdminFace(collection_id):
2     photo = "admin.jpg"
3
4     with open(photo, 'rb') as image:
5         response = client.index_faces(
6             CollectionId = collection_id,
7             Image = {
8                 'Bytes': image.read()
9             },
10            ExternalImageId = photo,
11            QualityFilter = "AUTO",
12            DetectionAttributes = ['ALL']
13        )
14    faceid = response['FaceRecords'][0]['Face']['FaceId']
15    print("Admin indexed successfully!")
16    return faceid
```

Listing 5: fracs-init.py - "Super Admin" creation

Finally, the admin is stored in the local database to be able to interact with the desktop app.

```
1 def createAdminDBEntry(face_id):
2     db_connection = mysql.connector.connect(host = "localhost", user = "hoda",
3     password = "XXXX", database = "FracsDB")
4     db_cursor = db_connection.cursor()
5
6     p = "INSERT INTO fracs_users (firstname, lastname, login_username,
7     login_password, is_admin, face_id) VALUES ('hoda','ahmed','hdahedo','" +
8     pbkdf2_sha256.hash("efta7") + "', 1,'" + face_id + "');"
9     db_cursor.execute(p)
10    db_connection.commit()
11    db_connection.close()
12    print("Admin user created successfully! You may now login to the FRACS system
13    .")
```

Listing 6: fracs-init.py - Register Super Admin locally

2.2 Database Operations

The database operation python file is specified for only operations that require interaction with the local database. This can include login validation, registering a user or for logging. The following code snippet is placed at the top of the dbOperations.py file for the imports and global variables declaration.

```
1 import mysql.connector
2 from passlib.hash import pbkdf2_sha256
3
4 db_connection = mysql.connector.connect(host="localhost", user="hoda", password="
    OPENsesameFRACS2020", database="FracsDB")
5 db_cursor = db_connection.cursor()
```

Listing 7: dbOperations.py

The *getHashedPassword()* function takes in two parameters, a login username and a face id. A hashed password for the user trying to login is returned to check if the password they entered is correct.

```
1 def getHashedPassword(username, face_id):
2     query = "SELECT login_password from fracs_users where login_username = '" +
    username + "'AND face_id = '" + face_id + "';"
3     db_cursor.execute(query)
4     password = db_cursor.fetchone()
5     if password is not None:
6         return password[0]
7     return None
```

Listing 8: dbOperations.py - getHashedPassword()

The *getAttemptedUserId()* function returns the ID of the user whose login username has been entered in a login attempt. This is used for logging, and can be useful in the event of user trying to login as another user using their credentials. However, that login attempt will fail, as the credentials have to match with the face of the user attempting to login.

```
1 def getAttemptedUserId(username):
2     query = "SELECT user_id from fracs_users where login_username = '" + username
    + "';"
3     db_cursor.execute(query)
4     userid = db_cursor.fetchone()
5     if userid is not None:
6         return userid[0]
7     return -1
```

Listing 9: dbOperations.py - getAttemptedUserId()

The *logAccessEvent()* function logs every access attempt/every login attempt to FRACS. These logs can then be viewed by admins at a later stage on the web application.

```
1 def logAccessEvent(logInfo):
2     query = "INSERT INTO fracs_logs (username_entered, face_id, attempted_user_id
    , login_datetime, is_successful) VALUES ('" + logInfo["username_entered"] + "
    ','" + logInfo["face_id"] + "',' + str(logInfo["attempted_user_id"]) + "','
    +logInfo["login_datetime"] + "',' + logInfo["is_successful"] + "');"
3     print(query)
4     db_cursor.execute(query)
5     db_connection.commit()
6     db_connection.close()
```

Listing 10: dbOperations.py - logAccessEvent()

The *getAdminHashedPassword()* is similar to the above *getHashedPassword()* function, but is defined specifically for admins as the query is different.

```
1 def getAdminHashedPassword(username, face_id):
2     query = "SELECT login_password from fracs_users where login_username = '" +
3     username + "' AND face_id = '" + face_id + "' and is_admin = 1;"
4     db_cursor.execute(query)
5     password = db_cursor.fetchone()
6     if password is not None:
7         return password[0]
8     return None
```

Listing 11: dbOperations.py - getAdminHashedPassword()

The *getAdminPhoneNumber()* function is used for SMS verification (2FA) during the registration process.

```
1 def getAdminPhoneNumber(username):
2     query = "SELECT phone_number from fracs_users where login_username = '" +
3     username + "' and is_admin = 1;"
4     db_cursor.execute(query)
5     phone_num = db_cursor.fetchone()
6     if phone_num is not None:
7         return phone_num[0]
8     return None
```

Listing 12: dbOperations.py - getAdminPhoneNumber()

The *checkUsername()* function checks the availability of a username, and this would be invoked in the registration process when a user wished to use a certain username. If the username is not available for use, the user is notified of this.

```
1 def checkUsername(username):
2     query = "SELECT user_id from fracs_users where login_username = '" + username
3     + "';"
4     db_cursor.execute(query)
5     usernameTaken = db_cursor.fetchone()
6     if usernameTaken is not None:
7         return False
8     return True
```

Listing 13: dbOperations.py - checkUsername()

The *addNewUser()* function adds a new user entry in the database, allowing them to login in the future.

```
1 def addNewUser(newUserInfo):
2     query = "INSERT INTO fracs_users (firstname, lastname, login_username,
3     login_password, is_admin, face_id, admin_registered) VALUES ('" + newUserInfo
4     ["firstname"] + "', '" + newUserInfo["lastname"] + "', '" + newUserInfo["
5     login_username"] + "', '" + newUserInfo["login_password"] + "', '" + str(
6     newUserInfo["is_admin"]) + "', '" + newUserInfo["face_id"] + "', '" +
7     newUserInfo["admin_registered"] + "');"
8     print(query)
9     db_cursor.execute(query)
10    db_connection.commit()
11    db_connection.close()
12    print("User created successfully! You may now login to FRACS")
```

Listing 14: dbOperations.py - checkUsername()

2.3 AWS Rekognition Operations

The rekognition operation python file is specified for only operations that are carried out using AWS Rekognition API. This can include detecting faces in a picture, verifying whether a detected face is registered with the system or not and adding new faces to a specified collection.

In order for this code to run and for the "boto3" module to be imported by Python, the AWS credentials must be stored in the home directory of the computer.

```
1 import boto3
2 from botocore.exceptions import ClientError
3
4 collection_id = "FRACUsers"
5 client=boto3.client('rekognition')
```

Listing 15: rekognitionOperations.py

The *confirmUserFace()* function is used when a user/admin wishes to login.

```
1 def confirmUserFace():
2     fileName='user.jpg'
3     threshold = 70
4
5     '''
6     return codes:    0 = no face detected
7                     1 = face detected but unknown (not registered with fracs)
8                     -1 = more than one face detected
9     '''
10
11     imgFaceNum = detectFaces(fileName)
12     if (imgFaceNum == 0):
13         return 0
14     elif (imgFaceNum > 1):
15         return -1
16     else:
17         #opening file & getting face
18         with open(fileName, 'rb') as image:
19             response=client.search_faces_by_image(
20                 CollectionId=collection_id,
21                 Image=
22                     {
23                         'Bytes': image.read()
24                     },
25                 FaceMatchThreshold=threshold
26             )
27         # exception handling done here for "no face exists" scenario, but not for
28         # "more than one face exists" scenario
29         # there shouldn't be more than one user per face as per exception
30         # handling during registration
31
32         faceMatches=response['FaceMatches']
33         if faceMatches != []: #if list of faceMatches is not empty
34             return (faceMatches[0]['Face']['FaceId'])
35         return 1
```

Listing 16: rekognitionOperations.py - confirmUserFace()

The *detectFaces()* function examines a given image and detects the faces in that image.

```
1 def detectFaces(file):
2     fileName=file
3
4     #opening file & getting face
5     with open(fileName, 'rb') as image:
6         response=client.detect_faces(
7             Image=
8             {
9                 'Bytes': image.read()
10            }
11        )
12    return len(response['FaceDetails'])
13
14    # response contains a dictionary of faces detected & their details..
15    # if we get the length of that dictionary,
16    # we can determine the number of faces in the picture
```

Listing 17: rekognitionOperations.py - detectFaces()

The *addNewFace()* function adds a new face to the Rekognition collection. This function is invoked when a user is registering.

```
1 def addNewFace():
2     photo = "user.jpg"
3
4     with open(photo, 'rb') as image:
5         response=client.index_faces(
6             CollectionId=collection_id,
7             Image=
8             {
9                 'Bytes': image.read()
10            },
11            ExternalImageId = photo,
12            QualityFilter = "AUTO"
13        )
14
15     faceid = response['FaceRecords'][0]['Face']['FaceId']
16    return faceid
```

Listing 18: rekognitionOperations.py - addNewFace()

```
1 def main():
2     print("in rekOp")
3
4     if __name__ == '__main__':
5         main()
```

Listing 19: rekognitionOperations.py - main()

2.4 Twilio SMS API Operations

After logging in to register a user, the admin must use their phone number to verify themselves. To do this, the external system, Twilio, is used.

```
1 from twilio.rest import Client
2 from random import randint
3
4 smscode = 0
```

Listing 20: twilioOperations.py

The *sendSMSCode()* is called when the admin authenticates themselves and needs to complete two-factor authentication. Twilio sends a 6-digit code to the admin's phone number, after which they will need to enter that code into the system to verify using 2FA. It is also called when a new admin wishes to register their phone number with the system.

```
1 def sendSMSCode(recipient):
2     global smscode
3
4     range_start = 10**(6-1)
5     range_end = (10**6)-1
6     smscode = randint(range_start, range_end)
7
8     #Your Account SID from twilio.com/console
9     account_sid = "XXXXXX"
10
11    #Your Auth Token from twilio.com/console
12    auth_token = "XXXXXX"
13    client = Client(account_sid, auth_token)
14
15    message = client.messages.create(
16        to= recipient,
17        from_="XXXXXX", # TWILIO'S REGISTERED PHONE NUMBER
18        body=smscode
19    )
20    return True
```

Listing 21: twilioOperations.py - sendSMSCode()

The *AddNewNumber()* function is invoked when a new admin wishes to store their phone number information into the system to be able to register users at a later stage.

```
1
2 #will only work if twilio is upgraded/premium
3 def AddNewNumber(person, number):
4     #Your Account SID from twilio.com/console
5     account_sid = "XXXXXXXX"
6
7     #Your Auth Token from twilio.com/console
8     auth_token = "XXXXXXXX"
9     client = Client(account_sid, auth_token)
10
11    validation_request = client.validation_requests.create(
12        friendly_name=person,
13        phone_number=number
14    )
15
16    global smscode
17    smscode = validation_request.validation_code
18    return smscode
```

Listing 22: twilioOperations.py - AddNewNumber()

The `VerifyTwilioSMS()` is used when the system wishes to verify 2FA, after an admin/user enters the code they received on their mobile phones into the system to complete 2FA.

```
1 def VerifyTwilioSMS(code):
2     global smscode
3     enteredCode = int(code)
4
5     if (enteredCode == smscode):
6         return True
7     else:
8         return False
9
```

Listing 23: twilioOperations.py - VerifyTwilioSMS()

2.5 Lock Operation

The lock operation code is called when a user successfully logs in and is authorized to access the room. It is one file that consists of one block of code that carries out everything related to the solenoid lock.

```
1 import RPi.GPIO as GPIO
2 import time
3
4 '''
5 The GPIO.BOARD option specifies that you are referring to the pins by the number
6 of the pin the the plug - i.e
7 the numbers printed on the board (e.g. P1) and in the middle of the diagrams
8 below.
9
10 The GPIO.BCM option means that you are referring to the pins by the "Broadcom SOC
11 channel" number
12 '''
13
14 def unlock():
15     try:
16         GPIO.setmode(GPIO.BCM)
17         GPIO.setup(23, GPIO.OUT) #Connect LOCK to GPIO 23
18         GPIO.output(23, True)
19         time.sleep(6)
20         GPIO.output(23, False)
21         GPIO.cleanup()
22     except:
23         GPIO.cleanup()
24
25 def main():
26     unlock()
27
28 if __name__ == '__main__':
29     main()
30
```

Listing 24: lockOperation.py

2.6 Camera Operation

There is only one operation carried out by the camera - taking a picture. The *coverCounter()* function of the below code creates 3 overlays on the camera's preview and pauses the operation for 1 second, hence doing a countdown of 3 and then a picture is taken.

```
1 import picamera
2 from PIL import Image
3 from time import sleep
4
5 def coverCounter(camera):
6     # load the image
7     img1 = Image.open('images/cd3.png')
8     img2 = Image.open('images/cd2.png')
9     img3 = Image.open('images/cd1.png')
10
11     # create the pad
12     pad = Image.new('RGB', (
13         ((img1.size[0] + 31) // 32) * 32,
14         ((img1.size[1] + 15) // 16) * 16,
15     ))
16
17     # paste the overlay - 3
18     # display num 3 of countdown
19     pad.paste(img1, (0,0))
20     o = camera.add_overlay(pad.tobytes(), size=img1.size)
21     o.alpha = 128
22     o.layer = 3
23     sleep(1)
24
25     # Remove previous overlay (num 3)
26     camera.remove_overlay(o)
27
28     # paste the overlay - 2
29     # display num 2 of countdown
30     pad.paste(img2, (0,0))
31     o = camera.add_overlay(pad.tobytes(), size=img2.size)
32     o.alpha = 128
33     o.layer = 3
34
35     sleep(1)
36
37     #remove previous overlay - 2
38     camera.remove_overlay(o)
39
40     # paste the overlay - 1
41     # display num 1 of countdown
42     pad.paste(img3, (0,0))
43     o = camera.add_overlay(pad.tobytes(), size=img3.size)
44     o.alpha = 128
45     o.layer = 3
46     sleep(1)
47     camera.remove_overlay(o)
48
49 def main():
50     camera = picamera.PiCamera()
51     camera.resolution = (1280,1024)
52     camera.framerate = 24
53     camera.start_preview()
54     overlayCounter(camera)
55     camera.capture('user.jpg')
56     camera.stop_preview()
57     camera.close()
```

```
58
59 if __name__ == '__main__':
60     main()
```

Listing 25: camCaptureOperation.py

2.7 User Interface

2.7.1 Main UI

The UI of the FRACS desktop application was designed using Qt Designer, which is a piece of software that makes designing GUI designs for PyQt and Qt for C++ easier. It offers a wide variety of options for creating a design for the Qt application.

The following code is the UI code created by Qt Designer, used for the FRACS desktop application.

```
1  # -*- coding: utf-8 -*-
2
3  # Form implementation generated from reading ui file 'fracsUI.ui'
4  #
5  # Created by: PyQt5 UI code generator 5.14.0
6
7
8  from PyQt5 import QtCore, QtGui, QtWidgets
9
10
11 class Ui_MainWindow(object):
12     def setupUi(self, MainWindow):
13         MainWindow.setObjectName("MainWindow")
14         MainWindow.resize(801, 480)
15         font = QtGui.QFont()
16         font.setPointSize(14)
17         MainWindow.setFont(font)
18         self.centralwidget = QtWidgets.QWidget(MainWindow)
19         self.centralwidget.setObjectName("centralwidget")
20         self.stackedWidget = QtWidgets.QStackedWidget(self.centralwidget)
21         self.stackedWidget.setGeometry(QtCore.QRect(0, 0, 800, 480))
22         self.stackedWidget.setStyleSheet("background-image: url(./images/cyber5.
jpeg)")
23         self.stackedWidget.setObjectName("stackedWidget")
24         self.SPLASH = QtWidgets.QWidget()
25         self.SPLASH.setStyleSheet("background-image: url(./images/newcyber.gif);")
26
27         self.SPLASH.setObjectName("SPLASH")
28         self.label = QtWidgets.QLabel(self.SPLASH)
29         self.label.setGeometry(QtCore.QRect(20, 20, 341, 151))
30         font = QtGui.QFont()
31         font.setPointSize(50)
32         self.label.setFont(font)
33         self.label.setStyleSheet("color: rgb(255, 255, 255);\n"
"background-color: rgba(0,0,0, 0);\n"
"background-image: url(./images/transparent.png)")
34         self.label.setAlignment(QtCore.Qt.AlignCenter)
35         self.label.setWordWrap(True)
36         self.label.setObjectName("label")
37         self.stackedWidget.addWidget(self.SPLASH)
38         self.MENU = QtWidgets.QWidget()
39         self.MENU.setObjectName("MENU")
40         self.login = QtWidgets.QPushButton(self.MENU)
41         self.login.setGeometry(QtCore.QRect(110, 60, 571, 131))
42         font = QtGui.QFont()
43         font.setFamily("Ubuntu")
44         font.setPointSize(40)
45         font.setBold(False)
46         font.setItalic(False)
47         font.setWeight(50)
48         self.login.setFont(font)
49         self.login.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
50         self.login.setStyleSheet("background-color: rgb(30, 129, 196);\n"
```

```

52         "background-color: qlineargradient(spread:pad, x1:0, y1:0, x2:1, y2
:0, stop:0 rgba(69, 81, 136, 240), stop:1 rgba(255, 255, 255, 255));\n"
53         "background-color: rgb(238, 238, 238);\n"
54         "background-image: url(./images/transparent.png);\n"
55         "border-radius: 6px;\n"
56         "")
57     self.login.setObjectName("login")
58     self.registration = QtWidgets.QPushButton(self.MENU)
59     self.registration.setGeometry(QtCore.QRect(110, 270, 571, 141))
60     font = QtGui.QFont()
61     font.setPointSize(40)
62     self.registration.setFont(font)
63     self.registration.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
64     self.registration.setStyleSheet("background-color:rgb(30, 129, 196);\n"
65         "background-color: rgb(238, 238, 238);\n"
66         "background-image: url(./images/transparent.png);\n"
67         "border-radius: 6px;")
68     self.registration.setObjectName("registration")
69     self.stackedWidget.addWidget(self.MENU)
70     self.LOGIN = QtWidgets.QWidget()
71     self.LOGIN.setObjectName("LOGIN")
72     self.unameLabel = QtWidgets.QLabel(self.LOGIN)
73     self.unameLabel.setGeometry(QtCore.QRect(100, 150, 101, 21))
74     font = QtGui.QFont()
75     font.setPointSize(16)
76     self.unameLabel.setFont(font)
77     self.unameLabel.setStyleSheet("background-image: url(./images/transparent
.png);\n"
78         "color: rgb(255, 255, 255);")
79     self.unameLabel.setObjectName("unameLabel")
80     self.passLabel = QtWidgets.QLabel(self.LOGIN)
81     self.passLabel.setGeometry(QtCore.QRect(100, 270, 101, 21))
82     font = QtGui.QFont()
83     font.setPointSize(16)
84     self.passLabel.setFont(font)
85     self.passLabel.setStyleSheet("background-image: url(./images/transparent.
.png);\n"
86         "color: rgb(255, 255, 255);")
87     self.passLabel.setObjectName("passLabel")
88     self.usernameInput = QtWidgets.QLineEdit(self.LOGIN)
89     self.usernameInput.setGeometry(QtCore.QRect(100, 180, 591, 51))
90     self.usernameInput.setStyleSheet("background-color: rgb(255, 255, 255);\n
"
91         "background-image: url(./images/transparent.png);\n"
92         "border-radius: 6px;\n"
93         "")
94     self.usernameInput.setAlignment(QtCore.Qt.AlignCenter)
95     self.usernameInput.setObjectName("usernameInput")
96     self.passwordInput = QtWidgets.QLineEdit(self.LOGIN)
97     self.passwordInput.setGeometry(QtCore.QRect(100, 300, 591, 51))
98     self.passwordInput.setStyleSheet("background-color: rgb(255, 255, 255);\n
"
99         "background-image: url(./images/transparent.png);\n"
100         "border-radius: 6px;")
101     self.passwordInput.setEchoMode(QtWidgets.QLineEdit.Password)
102     self.passwordInput.setAlignment(QtCore.Qt.AlignCenter)
103     self.passwordInput.setObjectName("passwordInput")
104     self.goback = QtWidgets.QPushButton(self.LOGIN)
105     self.goback.setGeometry(QtCore.QRect(0, 0, 111, 21))
106     self.goback.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
107     self.goback.setStyleSheet("background-color: rgb(255, 255, 255);\n"
108         "background-image: url(./images/transparent.png);")
109     self.goback.setObjectName("goback")

```

```

110     self.loginTitle = QtWidgets.QLabel(self.LOGIN)
111     self.loginTitle.setGeometry(QtCore.QRect(200, 50, 381, 51))
112     font = QtGui.QFont()
113     font.setPointSize(30)
114     self.loginTitle.setFont(font)
115     self.loginTitle.setStyleSheet("background-image: url(./images/transparent
.png);\n"
116                                   "color: rgb(255, 255, 255);")
117     self.loginTitle.setObjectName("loginTitle")
118     self.loginBtn = QtWidgets.QPushButton(self.LOGIN)
119     self.loginBtn.setGeometry(QtCore.QRect(550, 400, 141, 41))
120     self.loginBtn.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
121     self.loginBtn.setStyleSheet("background-color: rgb(255, 255, 255);\n"
122                                 "background-image: url(./images/transparent.png);\n"
123                                 "border-radius: 6px;")
124     self.loginBtn.setObjectName("loginBtn")
125     self.stackedWidget.addWidget(self.LOGIN)
126     self.FACE_CAPTURE = QtWidgets.QWidget()
127     self.FACE_CAPTURE.setObjectName("FACE_CAPTURE")
128     self.camButton = QtWidgets.QPushButton(self.FACE_CAPTURE)
129     self.camButton.setGeometry(QtCore.QRect(80, 220, 601, 31))
130     font = QtGui.QFont()
131     font.setPointSize(20)
132     self.camButton.setFont(font)
133     self.camButton.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
134     self.camButton.setStyleSheet("background-image: url(./images/transparent.
.png);\n"
135                                   "border-radius:6px;\n"
136                                   "color: rgb(255, 255, 255);")
137     self.camButton.setObjectName("camButton")
138     self.goback_2 = QtWidgets.QPushButton(self.FACE_CAPTURE)
139     self.goback_2.setGeometry(QtCore.QRect(0, 0, 111, 21))
140     self.goback_2.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
141     self.goback_2.setStyleSheet("background-color: rgb(255, 255, 255);\n"
142                                 "background-image: url(./images/transparent.png);")
143     self.goback_2.setObjectName("goback_2")
144     self.stackedWidget.addWidget(self.FACE_CAPTURE)
145     self.LOGIN_SUCCESS = QtWidgets.QWidget()
146     self.LOGIN_SUCCESS.setObjectName("LOGIN_SUCCESS")
147     self.successTitle = QtWidgets.QLabel(self.LOGIN_SUCCESS)
148     self.successTitle.setGeometry(QtCore.QRect(170, 210, 441, 31))
149     font = QtGui.QFont()
150     font.setPointSize(20)
151     self.successTitle.setFont(font)
152     self.successTitle.setStyleSheet("background-image: url(./images/
transparent.png);\n"
153                                   "color: rgb(255, 255, 255);")
154     self.successTitle.setObjectName("successTitle")
155     self.stackedWidget.addWidget(self.LOGIN_SUCCESS)
156     self.LOGIN_FAIL = QtWidgets.QWidget()
157     self.LOGIN_FAIL.setObjectName("LOGIN_FAIL")
158     self.failTitle = QtWidgets.QLabel(self.LOGIN_FAIL)
159     self.failTitle.setGeometry(QtCore.QRect(190, 220, 391, 31))
160     font = QtGui.QFont()
161     font.setPointSize(20)
162     self.failTitle.setFont(font)
163     self.failTitle.setStyleSheet("background-image: url(./images/transparent.
.png);\n"
164                                   "color: rgb(255, 255, 255);")
165     self.failTitle.setObjectName("failTitle")
166     self.stackedWidget.addWidget(self.LOGIN_FAIL)
167     self.LOGIN_FAIL_ZERO = QtWidgets.QWidget()
168     self.LOGIN_FAIL_ZERO.setObjectName("LOGIN_FAIL_ZERO")

```

```

169     self.failTitleZero = QtWidgets.QLabel(self.LOGIN_FAIL_ZERO)
170     self.failTitleZero.setGeometry(QtCore.QRect(160, 200, 421, 101))
171     font = QtGui.QFont()
172     font.setPointSize(20)
173     self.failTitleZero.setFont(font)
174     self.failTitleZero.setStyleSheet("background-image: url(./images/
transparent.png);\n"
175         "color: rgb(255, 255, 255);")
176     self.failTitleZero.setAlignment(QtCore.Qt.AlignCenter)
177     self.failTitleZero.setWordWrap(True)
178     self.failTitleZero.setObjectName("failTitleZero")
179     self.stackedWidget.addWidget(self.LOGIN_FAIL_ZERO)
180     self.LOGIN_FAIL_UNAUTH = QtWidgets.QWidget()
181     self.LOGIN_FAIL_UNAUTH.setObjectName("LOGIN_FAIL_UNAUTH")
182     self.failTitleUnauth = QtWidgets.QLabel(self.LOGIN_FAIL_UNAUTH)
183     self.failTitleUnauth.setGeometry(QtCore.QRect(150, 190, 421, 101))
184     font = QtGui.QFont()
185     font.setPointSize(20)
186     self.failTitleUnauth.setFont(font)
187     self.failTitleUnauth.setStyleSheet("background-image: url(./images/
transparent.png);\n"
188         "color: rgb(255, 255, 255);")
189     self.failTitleUnauth.setAlignment(QtCore.Qt.AlignCenter)
190     self.failTitleUnauth.setWordWrap(True)
191     self.failTitleUnauth.setObjectName("failTitleUnauth")
192     self.stackedWidget.addWidget(self.LOGIN_FAIL_UNAUTH)
193     self.LOGIN_FAIL_TWO = QtWidgets.QWidget()
194     self.LOGIN_FAIL_TWO.setObjectName("LOGIN_FAIL_TWO")
195     self.failTitleUnauthTwo = QtWidgets.QLabel(self.LOGIN_FAIL_TWO)
196     self.failTitleUnauthTwo.setGeometry(QtCore.QRect(150, 160, 431, 111))
197     font = QtGui.QFont()
198     font.setPointSize(20)
199     self.failTitleUnauthTwo.setFont(font)
200     self.failTitleUnauthTwo.setStyleSheet("background-image: url(./images/
transparent.png);\n"
201         "color: rgb(255, 255, 255);")
202     self.failTitleUnauthTwo.setAlignment(QtCore.Qt.AlignCenter)
203     self.failTitleUnauthTwo.setWordWrap(True)
204     self.failTitleUnauthTwo.setObjectName("failTitleUnauthTwo")
205     self.failTitleUnauthTwoWarning = QtWidgets.QLabel(self.LOGIN_FAIL_TWO)
206     self.failTitleUnauthTwoWarning.setGeometry(QtCore.QRect(0, 430, 701, 51))
207     font = QtGui.QFont()
208     font.setPointSize(11)
209     self.failTitleUnauthTwoWarning.setFont(font)
210     self.failTitleUnauthTwoWarning.setStyleSheet("background-image: url(./
images/transparent.png);\n"
211         "color: rgb(252, 175, 62);")
212     self.failTitleUnauthTwoWarning.setAlignment(QtCore.Qt.AlignCenter)
213     self.failTitleUnauthTwoWarning.setWordWrap(True)
214     self.failTitleUnauthTwoWarning.setObjectName("failTitleUnauthTwoWarning")
215     self.stackedWidget.addWidget(self.LOGIN_FAIL_TWO)
216     self.SMS_VERIFICATION = QtWidgets.QWidget()
217     self.SMS_VERIFICATION.setObjectName("SMS_VERIFICATION")
218     self.smsLabel = QtWidgets.QLabel(self.SMS_VERIFICATION)
219     self.smsLabel.setGeometry(QtCore.QRect(100, 150, 551, 31))
220     font = QtGui.QFont()
221     font.setPointSize(16)
222     self.smsLabel.setFont(font)
223     self.smsLabel.setStyleSheet("background-image: url(./images/transparent.
png);\n"
224         "color: rgb(255, 255, 255);")
225     self.smsLabel.setObjectName("smsLabel")
226     self.smsCode = QtWidgets.QLineEdit(self.SMS_VERIFICATION)

```



```

227     self.smsCode.setGeometry(QtCore.QRect(210, 220, 361, 81))
228     self.smsCode.setStyleSheet("background-color: rgb(255, 255, 255);\n"
229                                "background-image: url(./images/transparent.png);\n"
230                                "border-radius: 6px;\n"
231                                "")
232     self.smsCode.setAlignment(QtCore.Qt.AlignCenter)
233     self.smsCode.setObjectName("smsCode")
234     self.smsVerify = QtWidgets.QPushButton(self.SMS_VERIFICATION)
235     self.smsVerify.setGeometry(QtCore.QRect(550, 400, 141, 41))
236     font = QtGui.QFont()
237     font.setPointSize(11)
238     self.smsVerify.setFont(font)
239     self.smsVerify.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
240     self.smsVerify.setStyleSheet("background-image: url(./images/transparent.
png);\n"
241                                  "")
242     self.smsVerify.setObjectName("smsVerify")
243     self.goback_3 = QtWidgets.QPushButton(self.SMS_VERIFICATION)
244     self.goback_3.setGeometry(QtCore.QRect(0, 0, 111, 21))
245     self.goback_3.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
246     self.goback_3.setStyleSheet("background-color: rgb(255, 255, 255);\n"
247                                 "background-image: url(./images/transparent.png);")
248     self.goback_3.setObjectName("goback_3")
249     self.stackedWidget.addWidget(self.SMS_VERIFICATION)
250     self.NEW_USER_FORM = QtWidgets.QWidget()
251     self.NEW_USER_FORM.setObjectName("NEW_USER_FORM")
252     self.newFirstnameLabel = QtWidgets.QLabel(self.NEW_USER_FORM)
253     self.newFirstnameLabel.setGeometry(QtCore.QRect(60, 90, 91, 20))
254     font = QtGui.QFont()
255     font.setPointSize(12)
256     self.newFirstnameLabel.setFont(font)
257     self.newFirstnameLabel.setStyleSheet("background-image: url(./images/
transparent.png);\n"
258                                           "color: rgb(255, 255, 255);\n"
259                                           "")
260     self.newFirstnameLabel.setObjectName("newFirstnameLabel")
261     self.userInfoBtn = QtWidgets.QPushButton(self.NEW_USER_FORM)
262     self.userInfoBtn.setGeometry(QtCore.QRect(600, 410, 141, 41))
263     self.userInfoBtn.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
264     self.userInfoBtn.setStyleSheet("background-image: url(./images/
transparent.png);")
265     self.userInfoBtn.setObjectName("userInfoBtn")
266     self.newFirstnameInput = QtWidgets.QLineEdit(self.NEW_USER_FORM)
267     self.newFirstnameInput.setGeometry(QtCore.QRect(60, 110, 681, 41))
268     self.newFirstnameInput.setStyleSheet("background-color: rgb(255, 255,
255);\n"
269                                           "background-image: url(./images/transparent.png);\n"
270                                           "border-radius: 6px;\n"
271                                           "")
272     self.newFirstnameInput.setAlignment(QtCore.Qt.AlignCenter)
273     self.newFirstnameInput.setObjectName("newFirstnameInput")
274     self.registerTitle = QtWidgets.QLabel(self.NEW_USER_FORM)
275     self.registerTitle.setGeometry(QtCore.QRect(230, 20, 341, 51))
276     font = QtGui.QFont()
277     font.setPointSize(30)
278     self.registerTitle.setFont(font)
279     self.registerTitle.setStyleSheet("background-image: url(./images/
transparent.png);\n"
280                                       "color: rgb(255, 255, 255);")
281     self.registerTitle.setObjectName("registerTitle")
282     self.newLastNameInput = QtWidgets.QLineEdit(self.NEW_USER_FORM)
283     self.newLastNameInput.setGeometry(QtCore.QRect(60, 210, 681, 41))

```

```

284         self.newLastnameInput.setStyleSheet("background-color: rgb(255, 255, 255)
;\n"
285         "background-image: url(./images/transparent.png);\n"
286         "border-radius: 6px;\n"
287         "")
288         self.newLastnameInput.setEchoMode(QtWidgets.QLineEdit.Normal)
289         self.newLastnameInput.setAlignment(QtCore.Qt.AlignCenter)
290         self.newLastnameInput.setObjectName("newLastnameInput")
291         self.newLastnameLabel = QtWidgets.QLabel(self.NEW_USER_FORM)
292         self.newLastnameLabel.setGeometry(QtCore.QRect(60, 190, 81, 20))
293         font = QtGui.QFont()
294         font.setPointSize(12)
295         self.newLastnameLabel.setFont(font)
296         self.newLastnameLabel.setStyleSheet("background-image: url(./images/
transparent.png);\n"
297         "color: rgb(255, 255, 255);")
298         self.newLastnameLabel.setObjectName("newLastnameLabel")
299         self.newPassInput = QtWidgets.QLineEdit(self.NEW_USER_FORM)
300         self.newPassInput.setGeometry(QtCore.QRect(60, 410, 411, 41))
301         self.newPassInput.setStyleSheet("background-color: rgb(255, 255, 255);\n"
302         "background-image: url(./images/transparent.png);\n"
303         "border-radius: 6px;\n"
304         "")
305         self.newPassInput.setEchoMode(QtWidgets.QLineEdit.Password)
306         self.newPassInput.setAlignment(QtCore.Qt.AlignCenter)
307         self.newPassInput.setObjectName("newPassInput")
308         self.newUsernameLabel = QtWidgets.QLabel(self.NEW_USER_FORM)
309         self.newUsernameLabel.setGeometry(QtCore.QRect(60, 286, 121, 20))
310         font = QtGui.QFont()
311         font.setPointSize(12)
312         self.newUsernameLabel.setFont(font)
313         self.newUsernameLabel.setStyleSheet("background-image: url(./images/
transparent.png);\n"
314         "color: rgb(255, 255, 255);")
315         self.newUsernameLabel.setObjectName("newUsernameLabel")
316         self.newUsernameInput = QtWidgets.QLineEdit(self.NEW_USER_FORM)
317         self.newUsernameInput.setGeometry(QtCore.QRect(60, 310, 681, 41))
318         self.newUsernameInput.setStyleSheet("background-color: rgb(255, 255, 255)
;\n"
319         "background-image: url(./images/transparent.png);\n"
320         "border-radius: 6px;\n"
321         "")
322         self.newUsernameInput.setAlignment(QtCore.Qt.AlignCenter)
323         self.newUsernameInput.setObjectName("newUsernameInput")
324         self.newPassLabel = QtWidgets.QLabel(self.NEW_USER_FORM)
325         self.newPassLabel.setGeometry(QtCore.QRect(60, 386, 121, 20))
326         font = QtGui.QFont()
327         font.setPointSize(12)
328         self.newPassLabel.setFont(font)
329         self.newPassLabel.setStyleSheet("background-image: url(./images/
transparent.png);\n"
330         "color: rgb(255, 255, 255);")
331         self.newPassLabel.setObjectName("newPassLabel")
332         self.goback_4 = QtWidgets.QPushButton(self.NEW_USER_FORM)
333         self.goback_4.setGeometry(QtCore.QRect(0, 0, 111, 21))
334         self.goback_4.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
335         self.goback_4.setStyleSheet("background-color: rgb(255, 255, 255);\n"
336         "background-image: url(./images/transparent.png);\n"
337         "color: rgb(255, 255, 255);")
338         self.goback_4.setObjectName("goback_4")
339         self.isAdminRadio = QtWidgets.QRadioButton(self.NEW_USER_FORM)
340         self.isAdminRadio.setGeometry(QtCore.QRect(490, 410, 91, 41))
341         font = QtGui.QFont()
342         font.setPointSize(11)

```

```

342     self.isAdminRadio.setFont(font)
343     self.isAdminRadio.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
344     self.isAdminRadio.setStyleSheet("background-color: rgb(255, 255, 255);\n"
345                                     "background-image: url(./images/transparent.png);\n"
346                                     "border-radius: 6px;")
347     self.isAdminRadio.setObjectName("isAdminRadio")
348     self.stackedWidget.addWidget(self.NEW_USER_FORM)
349     self.NEW_ADMIN_FORM = QtWidgets.QWidget()
350     self.NEW_ADMIN_FORM.setObjectName("NEW_ADMIN_FORM")
351     self.newSMSCodeInput = QtWidgets.QLineEdit(self.NEW_ADMIN_FORM)
352     self.newSMSCodeInput.setGeometry(QtCore.QRect(40, 250, 481, 91))
353     self.newSMSCodeInput.setStyleSheet("background-color: rgb(255, 255, 255)
;\n"
354                                     "background-image: url(./images/transparent.png);\n"
355                                     "border-radius: 6px;\n"
356                                     "")
357     self.newSMSCodeInput.setAlignment(QtCore.Qt.AlignCenter)
358     self.newSMSCodeInput.setObjectName("newSMSCodeInput")
359     self.goback_5 = QtWidgets.QPushButton(self.NEW_ADMIN_FORM)
360     self.goback_5.setGeometry(QtCore.QRect(0, 0, 111, 21))
361     self.goback_5.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
362     self.goback_5.setStyleSheet("background-color: rgb(255, 255, 255);\n"
363                                 "background-image: url(./images/transparent.png);")
364     self.goback_5.setObjectName("goback_5")
365     self.newSMSVerify = QtWidgets.QPushButton(self.NEW_ADMIN_FORM)
366     self.newSMSVerify.setGeometry(QtCore.QRect(570, 270, 141, 51))
367     font = QtGui.QFont()
368     font.setPointSize(11)
369     self.newSMSVerify.setFont(font)
370     self.newSMSVerify.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
371     self.newSMSVerify.setStyleSheet("background-color: rgb(190, 206, 221);\n"
372                                     "background-image: url(./images/transparent.png);")
373     self.newSMSVerify.setObjectName("newSMSVerify")
374     self.newNumberInfo = QtWidgets.QTextEdit(self.NEW_ADMIN_FORM)
375     self.newNumberInfo.setGeometry(QtCore.QRect(40, 30, 741, 81))
376     self.newNumberInfo.setStyleSheet("background-color: rgb(190, 206, 221);\n"
"
377                                     "color: rgb(52, 101, 164);")
378     self.newNumberInfo.setObjectName("newNumberInfo")
379     self.newPhoneNumberInput = QtWidgets.QLineEdit(self.NEW_ADMIN_FORM)
380     self.newPhoneNumberInput.setGeometry(QtCore.QRect(40, 130, 481, 51))
381     self.newPhoneNumberInput.setStyleSheet("background-color: rgb(255, 255,
255);\n"
382                                     "background-image: url(./images/transparent.png);\n"
383                                     "border-radius: 6px;\n"
384                                     "")
385     self.newPhoneNumberInput.setText("")
386     self.newPhoneNumberInput.setMaxLength(14)
387     self.newPhoneNumberInput.setAlignment(QtCore.Qt.AlignCenter)
388     self.newPhoneNumberInput.setObjectName("newPhoneNumberInput")
389     self.newSendSMSCode = QtWidgets.QPushButton(self.NEW_ADMIN_FORM)
390     self.newSendSMSCode.setGeometry(QtCore.QRect(570, 130, 141, 51))
391     font = QtGui.QFont()
392     font.setPointSize(11)
393     self.newSendSMSCode.setFont(font)
394     self.newSendSMSCode.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor)
)
395     self.newSendSMSCode.setStyleSheet("background-color: rgb(190, 206, 221);\n"
n"
396                                     "background-image: url(./images/transparent.png);")
397     self.newSendSMSCode.setObjectName("newSendSMSCode")
398     self.newNumberSkipWarning = QtWidgets.QTextEdit(self.NEW_ADMIN_FORM)
399     self.newNumberSkipWarning.setGeometry(QtCore.QRect(40, 360, 741, 51))

```

```

400 self.newNumberSkipWarning.setStyleSheet("color: rgb(164, 0, 0);\n"
401      "background-color: rgb(252, 175, 62);")
402 self.newNumberSkipWarning.setObjectName("newNumberSkipWarning")
403 self.skipNewNumber = QtWidgets.QPushButton(self.NEW_ADMIN_FORM)
404 self.skipNewNumber.setGeometry(QtCore.QRect(570, 420, 141, 51))
405 font = QtGui.QFont()
406 font.setPointSize(11)
407 self.skipNewNumber.setFont(font)
408 self.skipNewNumber.setCursor(QtGui.QCursor(QtCore.Qt.PointingHandCursor))
409 self.skipNewNumber.setStyleSheet("background-color: rgb(164, 0, 0);\n"
410      "background-color: rgb(252, 175, 62);\n"
411      "background-image: url(./images/transparent.png);")
412 self.skipNewNumber.setObjectName("skipNewNumber")
413 self.stackedWidget.addWidget(self.NEW_ADMIN_FORM)
414 self.REGISTER_SUCCESS = QtWidgets.QWidget()
415 self.REGISTER_SUCCESS.setObjectName("REGISTER_SUCCESS")
416 self.regSuccess = QtWidgets.QLabel(self.REGISTER_SUCCESS)
417 self.regSuccess.setGeometry(QtCore.QRect(50, 210, 681, 31))
418 font = QtGui.QFont()
419 font.setPointSize(17)
420 self.regSuccess.setFont(font)
421 self.regSuccess.setStyleSheet("background-image: url(./images/transparent
422 .png);\n"
423      "color: rgb(255, 255, 255);")
424 self.regSuccess.setObjectName("regSuccess")
425 self.stackedWidget.addWidget(self.REGISTER_SUCCESS)
426 self.REGISTER_FAIL = QtWidgets.QWidget()
427 self.REGISTER_FAIL.setObjectName("REGISTER_FAIL")
428 self.regFail = QtWidgets.QLabel(self.REGISTER_FAIL)
429 self.regFail.setGeometry(QtCore.QRect(250, 210, 291, 41))
430 font = QtGui.QFont()
431 font.setPointSize(24)
432 self.regFail.setFont(font)
433 self.regFail.setStyleSheet("background-image: url(./images/transparent.
434 png);\n"
435      "color: rgb(255, 255, 255);")
436 self.regFail.setObjectName("regFail")
437 self.stackedWidget.addWidget(self.REGISTER_FAIL)
438 MainWindow.setCentralWidget(self.centralwidget)
439
440 self.retranslateUi(MainWindow)
441 self.stackedWidget.setCurrentIndex(11)
442 QtCore.QMetaObject.connectSlotsByName(MainWindow)
443
444 def retranslateUi(self, MainWindow):
445     _translate = QtCore.QCoreApplication.translate
446     MainWindow.setWindowTitle(_translate("MainWindow", "FRACS"))
447     self.label.setText(_translate("MainWindow", "WELCOME to FRACS"))
448     self.login.setText(_translate("MainWindow", "Login"))
449     self.registration.setText(_translate("MainWindow", "Registration"))
450     self.unameLabel.setText(_translate("MainWindow", "Username:"))
451     self.passLabel.setText(_translate("MainWindow", "Password:"))
452     self.usernameInput.setPlaceholderText(_translate("MainWindow", "Enter
453 Username"))
454     self.passwordInput.setPlaceholderText(_translate("MainWindow", "Enter
455 Password"))
456     self.goback.setText(_translate("MainWindow", "Main Menu"))
457     self.loginTitle.setText(_translate("MainWindow", "Login to access room"))
458     self.loginBtn.setText(_translate("MainWindow", "Next"))
459     self.camButton.setText(_translate("MainWindow", "Click here to start
460 camera, when you\'re ready!"))
461     self.goback_2.setText(_translate("MainWindow", "Main Menu"))

```

```

457         self.successTitle.setText(_translate("MainWindow", "Login Successful :D
Door unlocking.."))
458         self.failTitle.setText(_translate("MainWindow", "Login Failed :( Please
try again.."))
459         self.failTitleZero.setText(_translate("MainWindow", "Login Failed :( No
face detected. Please stand in front of camera"))
460         self.failTitleUnauth.setText(_translate("MainWindow", "Login Failed :(
User unrecognized. Are you sure you have registered with the FRACS system?"))
461         self.failTitleUnauthTwo.setText(_translate("MainWindow", "Login Failed :(
More than one face detected. Please ensure that only you stand in fron of
the camera"))
462         self.failTitleUnauthTwoWarning.setText(_translate("MainWindow", "*Note
that unregistered users are not allowed to access room. If other person(s) is
/are unregistered, you will be responsible if they gain access to room"))
463         self.smsLabel.setText(_translate("MainWindow", "Admin, enter the 6 digit
code that was sent to your phone:"))
464         self.smsCode.setPlaceholderText(_translate("MainWindow", "Enter SMS Code"
))
465         self.smsVerify.setText(_translate("MainWindow", "Verify"))
466         self.goback_3.setText(_translate("MainWindow", "Main Menu"))
467         self.newFirstnameLabel.setText(_translate("MainWindow", "First Name:"))
468         self.userInfoBtn.setText(_translate("MainWindow", "Next"))
469         self.newFirstnameInput.setPlaceholderText(_translate("MainWindow", "Enter
First Name"))
470         self.registerTitle.setText(_translate("MainWindow", "Register to FRACS"))
471         self.newLastnameInput.setPlaceholderText(_translate("MainWindow", "Enter
Last Name"))
472         self.newLastnameLabel.setText(_translate("MainWindow", "Last Name:"))
473         self.newPassInput.setPlaceholderText(_translate("MainWindow", "Enter
Password"))
474         self.newUsernameLabel.setText(_translate("MainWindow", "Login Username:"))
475         self.newUsernameInput.setPlaceholderText(_translate("MainWindow", "Enter
Username"))
476         self.newPassLabel.setText(_translate("MainWindow", "Login Password:"))
477         self.goback_4.setText(_translate("MainWindow", "Main Menu"))
478         self.isAdminRadio.setText(_translate("MainWindow", " Admin?"))
479         self.newSMSCodeInput.setPlaceholderText(_translate("MainWindow", "Enter
SMS Code"))
480         self.goback_5.setText(_translate("MainWindow", "Main Menu"))
481         self.newSMSVerify.setText(_translate("MainWindow", "Verify + finish!"))
482         self.newNumberInfo.setHtml(_translate("MainWindow", "<!DOCTYPE HTML
PUBLIC "-//W3C//DTD HTML 4.0//EN" "http://www.w3.org/TR/REC-html40/strict.
dtd">\n"
483             "<html><head><meta name=\"qrichtext\" content=\"1\" /><style type=\"
text/css\">\n"
484             "p, li { white-space: pre-wrap; }\n"
485             "</style></head><body style=\" font-family:\'Ubuntu\'; font-size:11pt
; font-weight:400; font-style:normal;\n">\n"
486             "<p align=\"center\" style=\" margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\n"><
span style=\" font-size:14pt;\n">Please enter your phone number in the below
field, then click \'Send Code\'. </span></p>\n"
487             "<p align=\"center\" style=\" margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\n"><
span style=\" font-size:14pt;\n">You will get a 6-digit code to that phone
number, which you will need to enter below.</span></p></body></html>"))
488         self.newPhoneNumberInput.setPlaceholderText(_translate("MainWindow",
"Enter Phone Number in this format: +353xxxxxxxx"))
489         self.newSendSMSCode.setText(_translate("MainWindow", "Send Code"))
490         self.newNumberSkipWarning.setHtml(_translate("MainWindow", "<!DOCTYPE
HTML PUBLIC "-//W3C//DTD HTML 4.0//EN" "http://www.w3.org/TR/REC-html40/
strict.dtd">\n"

```



```

491         "<html><head><meta name=\"qrichtext\" content=\"1\" /><style type=\"
text/css\">\n"
492         "p, li { white-space: pre-wrap; }\n"
493         "</style></head><body style=\" font-family:\'Ubuntu\'; font-size:11pt
; font-weight:400; font-style:normal;\">>\n"
494         "<p align=\"center\" style=\" margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\">>Don
\'t have your phone? You can skip this part and do it on the FRACS Website
later. </p>\n"
495         "<p align=\"center\" style=\" margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\">>
However, you will not be able to register new users until you verify your
phone number!</p></body></html>"))
496         self.skipNewNumber.setText(_translate("MainWindow", "Skip"))
497         self.regSuccess.setText(_translate("MainWindow", "Registration Successful
! You can now gain access by logging in :D"))
498         self.regFail.setText(_translate("MainWindow", "Registration Failed.."))

```

Listing 26: fracsUI.py

2.7.2 Pages Fading UI

Besides the main UI used for the FRACS desktop application, there is a code snippet that was used to perform a fading effect between pages of the application. This snippet of code was written and uploaded by user DavidBoddie on the official Python Wiki website.¹

```

1  import sys
2  from PyQt5.QtCore import QTimeLine
3  from PyQt5.QtGui import *
4  from PyQt5 import QtWidgets, QtCore, QtGui
5  from PyQt5.QtWidgets import QApplication, QMainWindow, QPushButton, QWidget
6
7  class FaderWidget(QWidget):
8      def __init__(self, old_widget, new_widget):
9          QWidget.__init__(self, new_widget)
10         self.old_pixmap = QtGui.QPixmap(new_widget.size())
11         old_widget.render(self.old_pixmap)
12         self.pixmap_opacity = 1.0
13
14         self.timeline = QTimeLine()
15         self.timeline.valueChanged.connect(self.animate)
16         self.timeline.finished.connect(self.close)
17         self.timeline.setDuration(333)
18         self.timeline.start()
19
20         self.resize(new_widget.size())
21         self.show()
22
23     def paintEvent(self, event):
24         painter = QPainter()
25         painter.begin(self)
26         painter.setOpacity(self.pixmap_opacity)
27         painter.drawPixmap(0, 0, self.old_pixmap)
28         painter.end()
29
30     def animate(self, value):
31         self.pixmap_opacity = 1.0 - value
32         self.repaint()

```

Listing 27: uifade.py

¹<https://wiki.python.org/moin/PyQt/Fading%20Between%20Widgets>

2.8 Main Code - fracs.py

The main code file, `fracs.py`, is the starting point of the whole FRACS application. It imports all the described files above and starts the application, carrying out various operations using the imported files.

The `main()` function in the file initializes the app and starts the GUI. The Qt application is contained in a class called `MainWindow()`

```
1
2 import os
3 import sys
4 import time
5
6 import fracsUI
7 import cameraCountdown
8 import rekognitionOperations
9 import dbOperations
10 import twilioOperations
11 import lockOperation
12
13 from pyfade import FaderWidget
14 from PyQt5 import QtWidgets, QtCore, QtGui
15 from PyQt5.QtCore import QTimer
16 from PyQt5.QtGui import QMovie, QPainter, QPixmap
17
18 from passlib.hash import pbkdf2_sha256
19
20 def main():
21     app = QtWidgets.QApplication(sys.argv)
22     win = MainWindow()
23     win.show()
24
25     app.exec_()
26
27 if __name__ == '__main__':
28     main()
```

Listing 28: `fracs.py` - `main()` - app initialization

The `init()` function is responsible for attaching the GUI components to the UI imported. It is also used for attaching button listeners and event handlers.

```
1 def __init__(self, parent=None):
2     #ui setup
3     super(MainWindow, self).__init__(parent)
4
5     self.ui = fracsUI.Ui_MainWindow()
6     self.ui.setupUi(self)
7
8     #attach button event listeners/handlers
9     self.attachButtonListeners()
10
11     #splash screen appears for 3 secs then redirects to main menu
12     QTimer.singleShot(3000, lambda: self.change(1))
```

Listing 29: `fracs.py` - `MainWindow.init()`

The `attachButtonListeners()` method attaches listeners to all buttons that exist in the application

```

1     def attachButtonListeners(self): #each button in the desktop app is assigned
a listener/handler
2     self.ui.stackedWidget.setCurrentIndex(0)
3     self.ui.login.clicked.connect(lambda: self.showLoginForm("USER"))
4     self.ui.loginBtn.clicked.connect(lambda: self.change(3))
5
6     self.ui.camButton.clicked.connect(self.openCamera)
7
8     self.ui.registration.clicked.connect(lambda: self.showLoginForm("ADMIN"))
9     self.ui.smsVerify.clicked.connect(self.verifyAdminSMS)
10
11    self.ui.userInfoBtn.clicked.connect(self.UsernameAvailability)
12    self.ui.newSendSMSCode.clicked.connect(self.verifyNewAdmin)
13    self.ui.skipNewNumber.clicked.connect(lambda: self.change(12))
14
15    #self.ui.newSMSVerify.clicked.connect(self.verifyNewAdminSMS) # will not work
!!!!!!! to register numbers using API, twilio must be upgraded!!!!
16
17    self.ui.goback.clicked.connect(self.restart)
18    self.ui.goback_2.clicked.connect(self.restart)
19    self.ui.goback_3.clicked.connect(self.restart)
20    self.ui.goback_4.clicked.connect(self.restart)
21    self.ui.goback_5.clicked.connect(self.restart)

```

Listing 30: fracs.py - MainWindow.init()

The *change()* function takes in a number, which indicates the index of the StackedWidget (the widget that controls the pages of the application). Each index points to a page in the StackedWidget, and when the *change()* function is called, the page that corresponds to the index appears.

The function also implements the fading between the pages.

```

1 def change(self, index):
2     self.fader_widget = FaderWidget(self.ui.stackedWidget.currentWidget(), self.
ui.stackedWidget.widget(index))
3     self.ui.stackedWidget.setCurrentIndex(index)

```

Listing 31: fracs.py - MainWindow.change()

The *showLoginForm()* function displays the login form depending on whether the intended action is login or register. If the role (declared as a global variable) is admin, the login form appears with a suitable message title, and if the role is user, the login form appears with another suitable message title.

```

1 def showLoginForm(self, role):
2     global approle
3     approle = role
4     if role == "ADMIN":
5         self.ui.loginTitle.setText("Admin Login for Registration")
6         self.ui.loginTitle.setGeometry(QtCore.QRect(120, 50, 500, 91))
7         self.change(2)
8     elif role == "USER":
9         self.ui.loginTitle.setText("Login to access room")
10        self.ui.loginTitle.setGeometry(QtCore.QRect(200, 50, 381, 51))
11        self.change(2)
12    else:
13        self.change(2)

```

Listing 32: fracs.py - MainWindow.showLoginForm()

The *openCamera()* function opens the camera page that informs the user that an image will be taken of their face, giving them a chance to appear in front of the camera and be prepared.

```
1 def openCamera(self):
2     cameraCountdown.main()
3     if approle == "NEW": # new user i.e. registration process
4         self.checkNewUserInfo()
5     else:
6         self.validateFaces(approle)
```

Listing 33: fracs.py - MainWindow.openCamera()

The *validateFaces()* function carries out more than one operation. It is responsible for ensuring that the image captured contains only one person, who is registered with the system. Here is also where the lock will be triggered if the user is authenticated and authorized to gain access.

It is responsible for logging all the login attempts and also for displaying success/fail messages to the screen.

```
1 def validateFaces(self, role):
2     face_id = rekognitionOperations.confirmUserFace()
3     '''
4     response codes: 0 = no face detected
5     1 = face detected but unknown (not registered with fracs)
6     -1 = more than one face detected
7     '''
8     if face_id is not None:
9         if face_id == 0: # no face detected
10             self.logEvent("ZERO", "FAIL")
11             self.change(6)
12             QTimer.singleShot(3000, self.restart)
13         elif face_id == -1:
14             self.logEvent("MANY", "FAIL")
15             self.change(8) # more than one face detected
16             QTimer.singleShot(3000, self.restart)
17         elif face_id == 1: # face detected, but unrecognized user
18             self.logEvent("UNKNOWN", "FAIL")
19             self.change(7)
20             QTimer.singleShot(3000, self.restart)
21
22     else:
23         valid = self.validateLogin(face_id)
24         if valid:
25             if role == "ADMIN":
26                 self.change(9)
27                 self.verifyAdmin()
28
29             elif role == "USER":
30                 self.change(4)
31                 self.unlock()
32
33             else:
34                 self.change(5)
35                 QTimer.singleShot(3000, self.restart)
36
37     else:
38         self.change(5)
39         QTimer.singleShot(3000, self.restart)
```

Listing 34: fracs.py - MainWindow.validateFaces()

The *logEvent()* function takes in parameters related to the login attempt and calls functions from the *dbOperations.py* file to do this.

```
1 def logEvent(self, faceid, status):
2     userInfo = {
3         "username_entered" : self.ui.usernameInput.text(),
4         "face_id" : faceid,
5         "attempted_user_id" : dbOperations.getAttemptedUserId(self.ui.
usernameInput.text()),
6         "login_datetime" : time.strftime('%Y-%m-%d %H:%M:%S'),
7         "is_successful" : status
8     }
9
10    dbOperations.logAccessEvent(userInfo)
```

Listing 35: *fracs.py* - *MainWindow.logEvent()*

The *unlock()* function triggers the *lockOperation.py* file, which in turn triggers the lock itself, allowing access into the room.

```
1 def unlock(self):
2     QTimer.singleShot(2000, lockOperation.main)
3     QTimer.singleShot(3000, self.restart)
```

Listing 36: *fracs.py* - *MainWindow.unlock()*

The *validateLogin()* function is not to be confused with the *validateFaces()* function. This function is mainly responsible for ensuring that the password entered by the user matches the password stored in the local database and is correct, whereas the *validateFaces()* function is responsible for the biometric aspect of authentication.

This function takes place depending on the output of the *validateFaces()* function.

```
1 def validateLogin(self, face_id):
2     global approle
3     hashedPassword = ""
4
5     if approle == "ADMIN":
6         hashedPassword = dbOperations.getAdminHashedPassword(self.ui.
usernameInput.text(), face_id)
7     elif approle == "USER":
8         hashedPassword = dbOperations.getHashedPassword(self.ui.usernameInput.
text(), face_id)
9
10    if hashedPassword is not None:
11        if pbkdf2_sha256.verify(self.ui.passwordInput.text(), hashedPassword): #
credentials match with face
12            self.logEvent(face_id, "SUCCESS")
13            return True
14        else:
15            self.logEvent(face_id, "FAIL")
16            return False
17    else:
18        self.logEvent(face_id, "FAIL")
19        return False
```

Listing 37: *fracs.py* - *MainWindow.init()*

The following two functions go hand-in-hand for ensuring that the admin verification process is correct.

```
1 def verifyAdmin(self):
2     adminPhoneNum = dbOperations.getAdminPhoneNumber(self.ui.usernameInput.text())
3     if adminPhoneNum is not None:
4         msgSent = twilioOperations.sendSMSCode(adminPhoneNum)
5     else:
6         ShowError("invalid admin phone number/issue")
7
8 def verifyAdminSMS(self):
9     success = twilioOperations.VerifyTwilioSMS(self.ui.smsCode.text())
10    if success:
11        self.logEvent()
12        self.change(10)
13    else:
14        self.ShowError("invalid code entered.")
```

Listing 38: fracs.py - MainWindow.verifyAdmin&SMS()

The *checkNewUserInfo()* function takes care of registering new users to the system by inserting their data in the local database.

```
1 def checkNewUserInfo(self):
2     is_admin = 0
3
4     if self.ui.isAdminRadio.isChecked():
5         is_admin = 1
6
7     faceAvailable = rekognitionOperations.confirmUserFace()
8
9     if faceAvailable == 1:
10        newFaceId= rekognitionOperations.addNewFace()
11        newUser = {
12            "firstname": self.ui.newFirstnameInput.text(),
13            "lastname": self.ui.newLastnameInput.text(),
14            "login_username": self.ui.newUsernameInput.text(),
15            "login_password": pbkdf2_sha256.hash(self.ui.newPassInput.text()),
16            "is_admin" : is_admin,
17            "face_id" : newFaceId,
18            "admin_registered" : self.ui.usernameInput.text()
19        }
20        dbOperations.addNewUser(newUser)
21        if self.ui.isAdminRadio.isChecked():
22            self.change(11)
23        else:
24            self.change(12)
25            QTimer.singleShot(3000, self.restart)
26
27    else:
28        self.change(13)
29        QTimer.singleShot(4000, self.restart)
```

Listing 39: fracs.py - MainWindow.checkNewUserInfo()

The *UsernameAvailability()* function returns an error to the currently registering user, informing them that the username they wish to register with is not available.

```

1 def UsernameAvailability(self):
2     global approle
3
4     username = self.ui.newUsernameInput.text()
5     availability = dbOperations.checkUsername(username)
6     if not availability:
7         self.ShowError("Username entered is not available.")
8         return False
9     else:
10        approle = "NEW"
11        self.change(3)
12        return True

```

Listing 40: fracs.py - MainWindow.UsernameAvailability()

The following two functions do the same thing as the *verifyAdmin()* and *verifyAdminSMS()* functions described above, respectively. Except, the following two functions are specified for newly registered admins.

```

1 def verifyNewAdmin(self):
2     newAdminPhoneNum = self.ui.newPhoneNumberInput.text()
3     if newAdminPhoneNum is not None:
4         msgSent = twilioOperations.sendSMSCode(newAdminPhoneNum)
5     else:
6         ShowError("invalid admin phone number/issue")
7
8
9 def verifyNewAdminSMS(self):
10    success = twilioOperations.VerifyTwilioSMS(self.ui.newSMSCodeInput.text())
11    if success:
12        self.change(12)
13    else:
14        self.change(5)
15    QTimer.singleShot(3000, self.restart)

```

Listing 41: fracs.py - MainWindow.verifyNewAdmin&SMS()

Very simply, the *showError()* function displays a warning dialog with the error message, which they can close down and modify/change their provided values.

```

1 def ShowError(self, errorMsg):
2     QtWidgets.QMessageBox.warning(self, 'Error', errorMsg)

```

Listing 42: fracs.py - MainWindow.showError()

Finally, *restart()* is responsible for restarting the application. It does this by clearing all text inputs in the application and deleting the image captured at the start permanently.

```

1 def restart(self):
2     self.ui.passwordInput.clear()
3     self.ui.usernameInput.clear()
4     self.ui.smsCode.clear()
5     self.ui.newFirstnameInput.clear()
6     self.ui.newLastnameInput.clear()
7     self.ui.newPassInput.clear()
8     self.ui.newUsernameInput.clear()
9     self.ui.newPhoneNumberInput.clear()
10    self.ui.newSMSCodeInput.clear()
11    self.ui.isAdminRadio.setChecked(False)
12    self.change(0)
13    os.remove("user.jpg") # deletes image, so that no PII remain on the system.
14    QTimer.singleShot(3000, lambda: self.change(1))

```

Listing 43: fracs.py - MainWindow.restart()

3 Web Application (Flask Webserver)

For Flask to work properly, there has to be a certain directory structure put in place. The "static" and "templates" folders have to exist for Flask to identify. The directory structure is as follows:

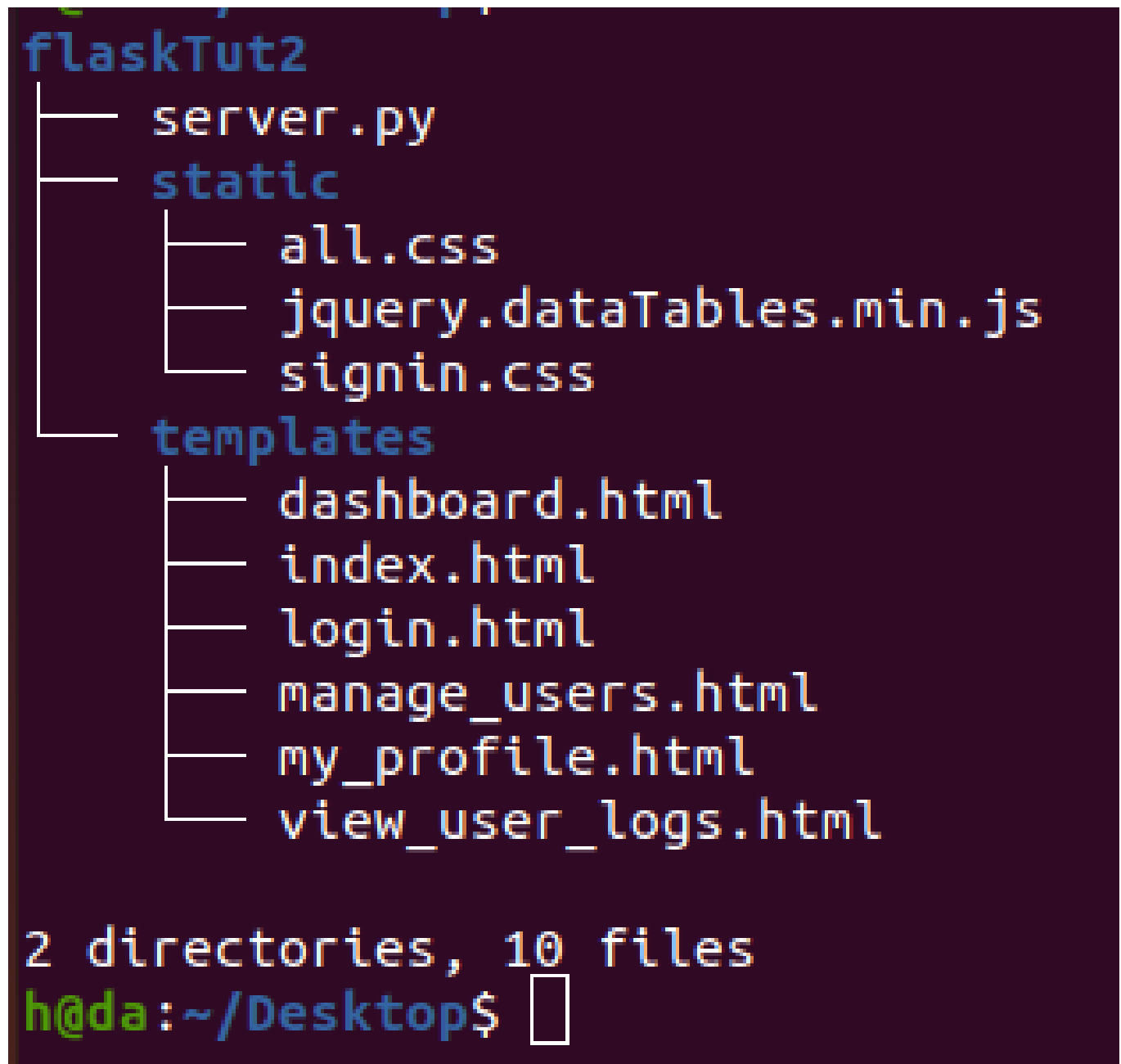


Figure 1: Web App Directory Structure

The main server file that contains all the webpages configuration and login security is called server.py. It does as follows:

```
1 from flask import Flask, render_template, redirect, url_for
2 from flask_bootstrap import Bootstrap
3 from flask_wtf import FlaskForm
4 from wtforms import StringField, PasswordField, BooleanField
5 from wtforms.validators import InputRequired, Length
6 from flask_sqlalchemy import SQLAlchemy
7 from sqlalchemy import cast, Date, and_
8 from passlib.hash import pbkdf2_sha256
9 from flask_login import LoginManager, UserMixin, login_user, login_required,
   current_user, logout_user
10 from datetime import datetime, timedelta
11 from dateutil.relativedelta import relativedelta
12
13
14 app = Flask(__name__)
15 app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
16 app.config['SECRET_KEY'] = 'XXXXXXX'
17 app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql://hoda:XXXX@localhost/FracsDB'
18                                     #mysql://username:password@host/database
19
20 db = SQLAlchemy(app)
21 Bootstrap(app)
22 login_manager = LoginManager()
23 login_manager.init_app(app)
24 login_manager.login_view = 'login'
25
26 class FracsUsers(UserMixin, db.Model):
27     __tablename__ = "fracs_users"
28     user_id = db.Column(db.Integer, primary_key=True)
29     first_name = db.Column(db.String(255))
30     last_name = db.Column(db.String(255))
31     login_username = db.Column(db.String(255), unique=True)
32     login_password = db.Column(db.String(255))
33     is_admin = db.Column(db.Boolean)
34     phone_number = db.Column(db.String(15))
35     face_id = db.Column(db.String(255))
36     admin_registered = db.Column(db.String(255))
37
38 def get_id(self):
39     return self.user_id
40
41 class FracsLogs(UserMixin, db.Model):
42     __tablename__ = "fracs_logs"
43     log_id = db.Column(db.Integer, primary_key=True)
44     username_entered = db.Column(db.String(255), unique=True)
45     attempted_user_id = db.Column(db.Integer)
46     face_id = db.Column(db.String(255))
47     login_datetime = db.Column(db.DateTime)
48     is_successful = db.Column(db.String(10))
49
50 def generatePieChart(type):
51     today = datetime.now().replace(hour=0, minute=0, second=0, microsecond=0)
52     weekAgo=0
53     monthAgo=0
54
55     if type == "weekly":
56         weekAgo = (today - timedelta(days = 7)).replace(hour=0, minute=0, second
57 =0, microsecond=0) # .replace gets rid of miliseconds
58         result = FracsLogs.query.filter(and_(FracLogs.login_datetime <= today ,
59 FracsLogs.login_datetime >= weekAgo)).all()
```

```

58     else:
59         monthAgo = today+relativedelta(months=-1)
60         result = FracsLogs.query.filter(and_(FracLogs.login_datetime <= today ,
61         FracsLogs.login_datetime >= monthAgo)).all()
62
63         successfulAttempts = 0
64         failedAttempts = 0
65
66         for attempt in result:
67             if attempt.is_successful == "SUCCESS":
68                 successfulAttempts+=1
69             else:
70                 failedAttempts+=1
71
72         labels = [
73             'SUCCESS', 'FAIL'
74         ]
75
76         values = [
77             successfulAttempts, failedAttempts
78         ]
79
80         colors = [
81             "#46BFBD", "#F7464A"
82         ]
83
84         return zip(values, labels, colors)
85
86 @app.route('/pie')
87 def pie():
88     logs = FracsLogs.query.order_by(FracLogs.login_datetime.desc()).all()
89     return render_template('test2.html')
90
91 @login_manager.user_loader
92 def load_user(user_id):
93     return FracsUsers.query.get(int(user_id))
94
95 class LoginForm(FlaskForm):
96     username = StringField('Username', validators=[InputRequired(), Length(min=4,
97     max=15)])
98     password = PasswordField('Password', validators=[InputRequired()])
99     remember = BooleanField('Remember Me')
100
101 @app.route('/', methods=['GET', 'POST'])
102 def index():
103     if current_user.is_authenticated:
104         return redirect(url_for('dashboard'))
105     return render_template('index.html')
106
107 @app.route('/login', methods=['GET', 'POST'])
108 def login():
109     form = LoginForm()
110     if form.validate_on_submit(): #if form is submitted & all validators returned
111         true
112         user = FracsUsers.query.filter_by(login_username=form.username.data).
113         first() # fetch first result from results where the username in db = username
114         provided in form
115         if user: #if there IS a result
116             if pbkdf2_sha256.verify(form.password.data, user.login_password ): #
117             and password hashes match
118                 login_user(user, remember=form.remember.data)
119                 return redirect(url_for('dashboard'))

```

```

115     return render_template('login.html', form=form)
116
117 @app.route('/dashboard')
118 @login_required
119 def dashboard():
120     if isAdmin() > 0:
121         logs = FracsLogs.query.order_by(FracLogs.login_datetime.desc()).limit
122         (10).all()
123         pi1 = generatePieChart(type="weekly")
124         pi2 = generatePieChart(type="monthly")
125         return render_template('dashboard.html', name=current_user.login_username
126         , logs=logs, set=pi1, set2=pi2)
127
128     else:
129         return my_profile()
130
131 def isAdmin():
132     return FracsUsers.query.filter_by(login_username=current_user.login_username,
133     is_admin=1).count() #number of entries in table that have login_username=
134     current user and is_admin=1..... if user is admin, result must have 1 entry,
135     else 0.
136
137 @app.route('/my_profile', methods=['GET', 'POST'])
138 @login_required
139 def my_profile():
140     userProfile = FracsUsers.query.filter_by(login_username=current_user.
141     login_username).first()
142     return render_template('my_profile.html', userprofile=userProfile)
143
144 @app.route('/change_password')
145 @login_required
146 def change_password():
147     return render_template('change_password.html')
148
149 @app.route('/view_user_logs')
150 @login_required
151 def view_user_logs():
152     logs = FracsLogs.query.order_by(FracLogs.login_datetime.desc()).all()
153     return render_template('view_user_logs.html', userslogs=logs)
154
155 @app.route('/all_users')
156 @login_required
157 def manage_users():
158     users = FracsUsers.query.order_by(FracUsers.firstname).all()
159     return render_template('manage_users.html', allusers=users)
160
161 @app.route('/logout')
162 def logout():
163     logout_user()
164     return redirect(url_for('index'))
165
166 if __name__ == '__main__':
167     app.run(debug=True, host='0.0.0.0')
168 #host='0.0.0.0' means web app is accessible to any device on the network

```

Listing 44: server.py


```

1 {% extends "bootstrap/base.html" %}
2
3 {% block title %}
4     FRACS
5 {% endblock %}
6
7 {% block styles %}
8     {{super()}}
9     <link rel="stylesheet" href="{url_for('.static', filename='starter-template.
css')}">
10 {% endblock %}
11
12 {% block content %}
13     <nav class="navbar navbar-inverse navbar-fixed-top">
14         <div class="container">
15             <div class="navbar-header">
16                 <button type="button" class="navbar-toggle collapsed" data-toggle
="collapse" data-target="#navbar" aria-expanded="false" aria-controls="navbar
">
17                     <span class="sr-only">Toggle navigation</span>
18                     <span class="icon-bar"></span>
19                     <span class="icon-bar"></span>
20                     <span class="icon-bar"></span>
21                 </button>
22                 <a class="navbar-brand" href="#">PI FRACS</a>
23             </div>
24             <div id="navbar" class="collapse navbar-collapse">
25                 <ul class="nav navbar-nav">
26                     <li class="active"><a href="#">Home</a></li>
27                     <li><a href="{url_for('login')}">Login</a></li>
28                 </ul>
29             </div><!-- /.nav-collapse -->
30         </div>
31     </nav>
32     <div class="container">
33
34         <div class="starter-template">
35             <h1>Welcome to FRACS Web!</h1>
36             <p class="lead">Here you can control FRACS<br>And its users</p>
37         </div>
38
39         </div><!-- /.container -->
40 {% endblock %}

```

Listing 45: index.html

```

1 {% extends "bootstrap/base.html" %}
2 {% import "bootstrap/wtf.html" as wtf %}
3
4 {% block title %}
5     Login
6 {% endblock %}
7
8 {% block styles %}
9     {{super()}}
10     <link rel="stylesheet" href="{{url_for('.static', filename='signin.css')}}">
11 {% endblock %}
12
13 {% block content %}
14     <div class="container">
15         <form class="form-signin" method="POST" action="/login">
16             <h2 class="form-signin-heading"> Please sign in </h2>
17             {{ form.csrf_token }}
18             {{ form.hidden_tag() }}
19             {{ wtf.form_field(form.username) }}
20             {{ wtf.form_field(form.password) }}
21             {{ wtf.form_field(form.remember) }}
22
23             <!-- OR -----
24             {{wtf.quick_form}} -->
25             <input class="btn btn-primary" type="submit" value="Login">
26         </form>
27     </div>
28 {% endblock %}

```

Listing 46: login.html

```

1 {% extends "bootstrap/base.html" %}
2
3 {% block title %}
4     Dashboard
5 {% endblock %}
6
7 {% block styles %}
8     {{super()}}
9     <link rel="stylesheet" href="{%url_for('static', filename='all.css')%}">
10    <script src='https://cdnjs.cloudflare.com/ajax/libs/Chart.js/1.0.2/Chart.min.
11    js'></script>
12 {% endblock %}
13
14 {% block content %}
15     <!-- TOP NAV BEGIN !!!!!!!!!!!!!!!!!!!!! -->
16     <nav class="navbar navbar-inverse navbar-fixed-top">
17         <div class="container-fluid">
18             <div class="navbar-header">
19                 <button type="button" class="navbar-toggle collapsed" data-toggle
20                 ="collapse" data-target="#navbar" aria-expanded="false" aria-controls="navbar
21                 ">
22
23                     <span class="sr-only">Toggle navigation</span>
24                     <span class="icon-bar"></span>
25                     <span class="icon-bar"></span>
26                     <span class="icon-bar"></span>
27
28                 </button>
29                 <a class="navbar-brand" href="#">FRACS Web</a>
30             </div>
31             <div id="navbar" class="navbar-collapse collapse">
32                 <ul class="nav navbar-nav navbar-right">
33                     <li><a href="/logout">Log Out</a></li>
34                 </ul>
35             </div>
36         </div>
37     </nav>
38
39     <div class="row">
40         <div class="col-md-12">
41             <div class="panel panel-default">
42                 <div class="panel-heading">
43                     <h3>FRACS Web</h3>
44                 </div>
45                 <div class="panel-body">
46                     <div class="row">
47                         <div class="col-md-6">
48                             <div class="panel panel-default">
49                                 <div class="panel-heading">
50                                     <h4>FRACS Web</h4>
51                                 </div>
52                                 <div class="panel-body">
53                                     <div class="row">
54                                         <div class="col-md-12">
55                                             <div class="panel panel-default">
56                                                 <div class="panel-heading">
57                                                     <h4>FRACS Web</h4>
58                                                 </div>
59                                                 <div class="panel-body">
60                                                     <div class="row">
61                                                         <div class="col-md-12">
62                                                             <div class="panel panel-default">
63                                                                 <div class="panel-heading">
64                                                                     <h4>FRACS Web</h4>
65                                                                 </div>
66                                                                 <div class="panel-body">
67                                                                     <div class="row">
68                                                                         <div class="col-md-12">
69                                                                             <div class="panel panel-default">
70                                                                                 <div class="panel-heading">
71                                                                                     <h4>FRACS Web</h4>
72                                                                                 </div>
73                                                                                 <div class="panel-body">
74                                                                                     <div class="row">
75                                                                                         <div class="col-md-12">
76                             </div>
77                         </div>
78                     </div>
79                 </div>
80             </div>
81         </div>
82     </div>
83
84     <div class="row">
85         <div class="col-md-12">
86             <div class="panel panel-default">
87                 <div class="panel-heading">
88                     <h3>FRACS Web</h3>
89                 </div>
90                 <div class="panel-body">
91                     <div class="row">
92                         <div class="col-md-6">
93                             <div class="panel panel-default">
94                                 <div class="panel-heading">
95                                     <h4>FRACS Web</h4>
96                                 </div>
97                                 <div class="panel-body">
98                                     <div class="row">
99                                         <div class="col-md-12">
100                                             <div class="panel panel-default">
101                                                 <div class="panel-heading">
102                                                     <h4>FRACS Web</h4>
103                                                 </div>
104                                                 <div class="panel-body">
105                                                     <div class="row">
106                                                         <div class="col-md-12">
107                                                             <div class="panel panel-default">
108                                                                 <div class="panel-heading">
109                                                                     <h4>FRACS Web</h4>
110                                                                 </div>
111                                                                 <div class="panel-body">
112                                                                     <div class="row">
113                                                                         <div class="col-md-12">
114                                                                             <div class="panel panel-default">
115                                                                                 <div class="panel-heading">
116                                                                                     <h4>FRACS Web</h4>
117                                                                                 </div>
118                                                                                 <div class="panel-body">
119                                                                                     <div class="row">
120                                                                                         <div class="col-md-12">
121                             </div>
122                         </div>
123                     </div>
124                 </div>
125             </div>
126         </div>
127     </div>
128
129     <div class="row">
130         <div class="col-md-12">
131             <div class="panel panel-default">
132                 <div class="panel-heading">
133                     <h3>FRACS Web</h3>
134                 </div>
135                 <div class="panel-body">
136                     <div class="row">
137                         <div class="col-md-6">
138                             <div class="panel panel-default">
139                                 <div class="panel-heading">
140                                     <h4>FRACS Web</h4>
141                                 </div>
142                                 <div class="panel-body">
143                                     <div class="row">
144                                         <div class="col-md-12">
145                                             <div class="panel panel-default">
146                                                 <div class="panel-heading">
147                                                     <h4>FRACS Web</h4>
148                                                 </div>
149                                                 <div class="panel-body">
150                                                     <div class="row">
151                                                         <div class="col-md-12">
152                                                             <div class="panel panel-default">
153                                                                 <div class="panel-heading">
154                                                                     <h4>FRACS Web</h4>
155                                                                 </div>
156                                                                 <div class="panel-body">
157                                                                     <div class="row">
158                                                                         <div class="col-md-12">
159                                                                             <div class="panel panel-default">
160                                                                                 <div class="panel-heading">
161                                                                                     <h4>FRACS Web</h4>
162                                                                                 </div>
163                                                                                 <div class="panel-body">
164                                                                                     <div class="row">
165                                                                                         <div class="col-md-12">
166                             </div>
167                         </div>
168                     </div>
169                 </div>
170             </div>
171         </div>
172     </div>
173
174     <div class="row">
175         <div class="col-md-12">
176             <div class="panel panel-default">
177                 <div class="panel-heading">
178                     <h3>FRACS Web</h3>
179                 </div>
180                 <div class="panel-body">
181                     <div class="row">
182                         <div class="col-md-6">
183                             <div class="panel panel-default">
184                                 <div class="panel-heading">
185                                     <h4>FRACS Web</h4>
186                                 </div>
187                                 <div class="panel-body">
188                                     <div class="row">
189                                         <div class="col-md-12">
190                                             <div class="panel panel-default">
191                                                 <div class="panel-heading">
192                                                     <h4>FRACS Web</h4>
193                                                 </div>
194                                                 <div class="panel-body">
195                                                     <div class="row">
196                                                         <div class="col-md-12">
197                                                             <div class="panel panel-default">
198                                                                 <div class="panel-heading">
199                                                                     <h4>FRACS Web</h4>
200                                                                 </div>
201                                                                 <div class="panel-body">
202                                                                     <div class="row">
203                                                                         <div class="col-md-12">
204                                                                             <div class="panel panel-default">
205                                                                                 <div class="panel-heading">
206                                                                                     <h4>FRACS Web</h4>
207                                                                                 </div>
208                                                                                 <div class="panel-body">
209                                                                                     <div class="row">
210                                                                                         <div class="col-md-12">
211                             </div>
212                         </div>
213                     </div>
214                 </div>
215             </div>
216         </div>
217     </div>
218
219     <div class="row">
220         <div class="col-md-12">
221             <div class="panel panel-default">
222                 <div class="panel-heading">
223                     <h3>FRACS Web</h3>
224                 </div>
225                 <div class="panel-body">
226                     <div class="row">
227                         <div class="col-md-6">
228                             <div class="panel panel-default">
229                                 <div class="panel-heading">
230                                     <h4>FRACS Web</h4>
231                                 </div>
232                                 <div class="panel-body">
233                                     <div class="row">
234                                         <div class="col-md-12">
235                                             <div class="panel panel-default">
236                                                 <div class="panel-heading">
237                                                     <h4>FRACS Web</h4>
238                                                 </div>
239                                                 <div class="panel-body">
240                                                     <div class="row">
241                                                         <div class="col-md-12">
242                                                             <div class="panel panel-default">
243                                                                 <div class="panel-heading">
244                                                                     <h4>FRACS Web</h4>
245                                                                 </div>
246                                                                 <div class="panel-body">
247                                                                     <div class="row">
248                                                                         <div class="col-md-12">
249                                                                             <div class="panel panel-default">
250                                                                                 <div class="panel-heading">
251                                                                                     <h4>FRACS Web</h4>
252                                                                                 </div>
253                                                                                 <div class="panel-body">
254                                                                                     <div class="row">
255                                                                                         <div class="col-md-12">
256                             </div>
257                         </div>
258                     </div>
259                 </div>
260             </div>
261         </div>
262     </div>
263
264     <div class="row">
265         <div class="col-md-12">
266             <div class="panel panel-default">
267                 <div class="panel-heading">
268                     <h3>FRACS Web</h3>
269                 </div>
270                 <div class="panel-body">
271                     <div class="row">
272                         <div class="col-md-6">
273                             <div class="panel panel-default">
274                                 <div class="panel-heading">
275                                     <h4>FRACS Web
```

```

30     </div>
31 </div>
32 </nav>
33 <!-- TOP NAV END !!!!!!!!!!!!!!!!!!!!!!! -->
34
35 <!-- SIDE NAV BEGIN !!!!!!!!!!!!!!!!!!!!!!! -->
36 <div class="container-fluid">
37     <div class="row">
38         <div class="col-sm-3 col-md-2 sidebar">
39             <ul class="nav nav-sidebar">
40                 <li class="active"><a href="#">Dashboard</a> <span class="sr-
only">(current)</span></li>
41                 <li><a href="/all_users">FRACS Users</a></li>
42                 <li><a href="/view_user_logs">View Access Logs </a></li>
43             </ul>
44             <ul class="nav nav-sidebar">
45                 <li><a href="/my_profile">User Profile</a></li>
46             </ul>
47         </div>
48
49 <!-- SIDE NAV END !!!!!!!!!!!!!!!!!!!!!!! -->
50
51         <div class="col-sm-9 col-sm-offset-3 col-md-10 col-md-offset-2 main">
52             <h1 class="page-header">Welcome, {{name}}!</h1>
53
54             <div class="row placeholders">
55                 <div class="col-xs-3 col-sm-6 placeholder">
56                     <canvas id="pieChart1" width="300" height="200"></canvas>
57                     <h4>Access Attempts</h4>
58                     <span class="text-muted">in previous week</span>
59                 </div>
60 <script>
61     var pieData = [
62         {% for item, label, colors in set %}
63         {
64             value: {{item}},
65             label: "{{label}}",
66             color : "{{colors}}"
67         },
68         {% endfor %}
69     ];
70
71     // draw pie pieChart1
72     new Chart(document.getElementById("pieChart1").getContext("2d")).Pie(pieData)
73 </script>
74
75                 <div class="col-xs-3 col-sm-6 placeholder">
76                     <canvas id="pieChart2" width="300" height="200"></canvas>
77
78                     <h4>Access Attempts</h4>
79                     <span class="text-muted">in previous month</span>
80                 </div>
81
82 <script>
83     var pie2Data = [
84         {% for item, label, colors in set2 %}
85         {
86             value: {{item}},
87             label: "{{label}}",
88             color : "{{colors}}"
89         },
90         {% endfor %}

```

```

91 ];
92
93 new Chart(document.getElementById("pieChart2").getContext("2d")).Pie(pie2Data
94 );
95 </script>
96
97 </div>
98
99 <!-- TABLE BEGIN!!!!!!!!!!!!!! -->
100 <h2 class="sub-header">Latest Access Logs</h2>
101 <div class="table-responsive">
102     <table class="table table-striped table-hover" border="4"
103     cellpadding="5" cellspacing="5" style="border-color: black;">
104         <thead>
105             <tr class="tbl-headings">
106                 <th>#</th>
107                 <th>Username Entered</th>
108                 <th>Actual User ID</th>
109                 <th>Date & Time</th>
110                 <th>Access Granted?</th>
111             </tr>
112         </thead>
113         <tbody>
114             {% for row in logs %}
115                 {% if row.is_successful == "FAIL" %}
116                 <tr class="danger">
117                     {% else %}
118                     <tr class="success">
119                     {% endif %}
120                     <td>{{ loop.index }}</td>
121                     <td>{{ row.username_entered }}</td>
122                     <td>{{ row.attempted_user_id }}</td>
123                     <td>{{ row.login_datetime }}</td>
124                     <td>{{ row.is_successful }}</td>
125                 </tr>
126             {% endfor %}
127         </tbody>
128     </table>
129 </div>
130 <!-- TABLE END!!!!!!!!!!!!!! -->
131 </div>
132 {% endblock %}

```

Listing 47: dashboard.html

```

1 {% extends "bootstrap/base.html" %}
2
3 {% block title %}
4 Dashboard
5 {% endblock %}
6
7 {% block styles %}
8 {{super()}}
9 <link rel="stylesheet" href="{%url_for('.static', filename='all.css')%}">
10 <script src='https://cdnjs.cloudflare.com/ajax/libs/Chart.js/1.0.2/Chart.min.js'>
11 </script>
12 {% endblock %}
13
14 {% block content %}
15 <!-- TOP NAV BEGIN !!!!!!!!!!!!!!!!!!!!!!! -->
16 <nav class="navbar navbar-inverse navbar-fixed-top">
17 <div class="container-fluid">
18 <div class="navbar-header">
19 <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-
20 -target="#navbar" aria-expanded="false" aria-controls="navbar">
21 <span class="sr-only">Toggle navigation</span>
22 <span class="icon-bar"></span>
23 <span class="icon-bar"></span>
24 <span class="icon-bar"></span>
25 </button>
26 <a class="navbar-brand" href="#">FRACS Web</a>
27 </div>
28 <div id="navbar" class="navbar-collapse collapse">
29 <ul class="nav navbar-nav navbar-right">
30 <li><a href="/logout">Log Out</a></li>
31 </ul>
32 </div>
33 </nav>
34 <!-- TOP NAV END !!!!!!!!!!!!!!!!!!!!!!! -->
35
36 <!-- SIDE NAV BEGIN !!!!!!!!!!!!!!!!!!!!!!! -->
37 <div class="container-fluid">
38 <div class="row">
39 <div class="col-sm-3 col-md-2 sidebar">
40 <ul class="nav nav-sidebar">
41 <li class="active"><a href="#">Dashboard</a> <span class="sr-only">(current)</
42 span></li>
43 <li><a href="/all_users">FRACS Users</a></li>
44 <li><a href="/view_user_logs">View Access Logs </a></li>
45 </ul>
46 <ul class="nav nav-sidebar">
47 <li><a href="/my_profile">User Profile</a></li>
48 </ul>
49 </div>
50 <!-- SIDE NAV END !!!!!!!!!!!!!!!!!!!!!!! -->
51
52 <!-- LOGS TABLE START !!!!!!!!!!!!!!!!!!!!!!! -->
53
54 <div class="col-sm-9 col-sm-offset-3 col-md-10 col-md-offset-2 main">
55 <h2 class="sub-header">FRACS Users</h2>
56 <div class="table-responsive">
57 <table id="myTable" class="table table-striped table-hover" border="1"
58 cellpadding="5" cellspacing="5" style="border-color: black;">
59 <thead>
60 <tr class="tbl-headings">

```

```

60 <th>ID</th>
61 <th>Name</th>
62 <th>Username</th>
63 <th>Admin?</th>
64 <th>Phone Number</th>
65 <th>Face ID</th>
66 <th>Admin Registered</th>
67 <th> </th>
68 </tr>
69 </thead>
70 <tbody >
71 {% for row in allusers %}
72 <td>{{ row.user_id}}</td>
73 <td>{{ row.firstname }} {{ row.lastname }}</td>
74 <td>{{ row.login_username }}</td>
75 <td>{{ row.is_admin }}</td>
76 <td>{{ row.phone_number }}</td>
77 <td>{{ row.face_id }}</td>
78 <td>{{ row.admin_registered }}</td>
79 <td><button type="button" data-toggle="modal" data-target="#myOutput{{row.user_id
    }}" class="btn btn-small btn-success btn-xs">?</button></td>
80 </tr>
81
82
83
84 {% endfor %}
85 </tbody>
86 </table>
87 <!-- LOGS TABLE END !!!!!!!!!!!!!!!!!!!!!!! -->
88
89 {% for row in allusers %}
90 <!-- Modal -->
91 <div class="modal fade" id="myOutput{{row.user_id}}" role="dialog">
92 <div class="modal-dialog modal-lg">
93
94 <!-- Modal content-->
95 <div class="modal-content">
96 <div class="modal-header">
97 <button type="button" class="close" data-dismiss="modal">&times;</button>
98 <h4 class="modal-title">Task Output</h4>
99 </div>
100 <div class="modal-body">
101 <div class="form-group">
102 <input type="text" class="form-control" value="Log ID: {{ row.user_id }}">
103 </div>
104 <div class="form-group">
105 <input type="text" class="form-control" value="Name: {{row.firstname}} {{row.
    lastname}}">
106 </div>
107 <div class="form-group">
108 <input type="text" class="form-control" value="Login Username: {{row.
    login_username}}" name="designation">
109 </div>
110 <div class="form-group">
111 <input type="text" class="form-control" value="Admin?: {{row.is_admin}}">
112 </div>
113 <div class="form-group">
114 <input type="text" class="form-control" value="Phone Number: {{row.phone_number}}
    ">
115 </div>
116 <div class="form-group">
117 <input type="text" class="form-control" value="Face ID: {{row.face_id}}">
118 </div>

```

```

119 <div class="form-group">
120 <input type="text" class="form-control" value="Admin Registered: {{row.
      admin_registered}}">
121 </div>
122
123 </div>
124 <div class="modal-footer">
125 <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
126 </div>
127 </div>
128
129 </div>
130 </div>
131 {% endfor %}
132
133 </div>
134 </div>
135 </div>
136 </div>
137 </div>
138 {% endblock %}

```

Listing 48: manage_users.html

```

1 {% extends "bootstrap/base.html" %}
2
3 {% block title %}
4 View User Logs
5 {% endblock %}
6
7 {% block head%}
8 {% block scripts %} <!-- SCRIPT BLOCK START -->
9
10 {{super()}}
11
12 <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js"></
    script> <!-- JQUERY - MUST BE FIRST -->
13 <script type="text/javascript" src="http://cdn.datatables.net/1.10.2/js/jquery.
    dataTables.min.js"></script> <!-- CDN DATATABLE SCRIPT -->
14
15 <script>
16 $(document).ready(function(){
17 $('#myTable').dataTable();
18 });
19 </script>
20
21 {% endblock %} <!-- SCRIPT BLOCK END -->
22
23 {% block styles %} <!-- STYLE BLOCK START -->
24 {{super()}}
25 <link rel="stylesheet" href="{url_for('.static', filename='all.css')}"> <!-- MY
    OWN CSS STYLING FILE -->
26 <link rel="stylesheet" href="http://cdn.datatables.net/1.10.2/css/jquery.
    dataTables.min.css"></style> <!-- CDN STYLING FILE -->
27 {% endblock %} <!-- STYLE BLOCK END -->
28
29 {% endblock %} <!-- BLOCK HEAD END -->
30
31
32 {% block content %}
33 <!-- TOP NAV BEGIN !!!!!!!!!!!!!!!!!!!!!!! -->
34 <nav class="navbar navbar-inverse navbar-fixed-top">
35 <div class="container-fluid">
36 <div class="navbar-header">
37 <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data
    -target="#navbar" aria-expanded="false" aria-controls="navbar">
38 <span class="sr-only">Toggle navigation</span>
39 <span class="icon-bar"></span>
40 <span class="icon-bar"></span>
41 <span class="icon-bar"></span>
42 </button>
43 <a class="navbar-brand" href="#">FRACS Web</a>
44 </div>
45 <div id="navbar" class="navbar-collapse collapse">
46 <ul class="nav navbar-nav navbar-right">
47 <li><a href="/logout">Log Out</a></li>
48 </ul>
49 </div>
50 </div>
51 </nav>
52 <!-- TOP NAV END !!!!!!!!!!!!!!!!!!!!!!! -->
53
54
55 <!-- SIDE NAV BEGIN !!!!!!!!!!!!!!!!!!!!!!! -->
56 <div class="container-fluid">
57 <div class="row">
58 <div class="col-sm-3 col-md-2 sidebar">

```



```

59 <ul class="nav nav-sidebar">
60 <li><a href="/dashboard">Dashboard</a></li>
61 <li><a href="/all_users">FRACS Users</a></li>
62 <li class="active"><a href="#">View Access Logs <span class="sr-only">(current)</span> </a></li>
63 </ul>
64 <ul class="nav nav-sidebar">
65 <li><a href="/my_profile">User Profile</a></li>
66 </ul>
67 </div>
68
69 <!-- SIDE NAV END !!!!!!!!!!!!!!!!!!!!!!! -->
70
71 <!-- LOGS TABLE START !!!!!!!!!!!!!!!!!!!!!!! -->
72
73 <div class="col-sm-9 col-sm-offset-3 col-md-10 col-md-offset-2 main">
74 <h2 class="sub-header">Access Logs</h2>
75 <div class="table-responsive">
76 <table id="myTable" class="table table-striped table-hover" border="1"
77     cellpadding="5" cellspacing="5" style="border-color: black;">
78 <thead>
79 <tr class="tbl-headings">
80 <th>#</th>
81 <th>Username Entered</th>
82 <th>Actual User ID</th>
83 <th>Date & Time</th>
84 <th>Access Granted?</th>
85 </tr>
86 </thead>
87 <tbody>
88 {% for row in userslogs %}
89 <p> {{ row.user_id}} </p>
90 {% if row.is_successful == "FAIL" %}
91 <tr class="danger">
92 {% else %}
93 <tr class="success">
94 {% endif %}
95 <td>{{ row.log_id}}</td>
96 <td>{{ row.username_entered }}</td>
97 <td>{{ row.attempted_user_id }} </td>
98 <td>{{ row.login_datetime }}</td>
99 <td>{{ row.is_successful }}</td>
100 </tr>
101 <!-- <tr style="background-color:#ff6666; border-color: #b30000">
102 <tr style="background-color:#99b3ff; border-color: #002699">
103 <td><a href="#" data-toggle="modal" data-target='#myModal' class="btn btn-success
104     ">?</a></td> -->
105 {% endfor %}
106 </tbody>
107 </table>
108 <!-- LOGS TABLE END !!!!!!!!!!!!!!!!!!!!!!! -->
109
110 </div>
111 </div>
112 </div>
113 </div>
114 </div>
115 {% endblock %}

```

Listing 49: view_user_logs.html

```

1 {% extends "bootstrap/base.html" %}
2
3 {% block title %}
4 User Profile
5 {% endblock %}
6
7 {% block styles %}
8 {{super()}}
9 <link rel="stylesheet" href="{%url_for('.static', filename='all.css')%}">
10 <link rel="import" href="navigation.html">
11
12 {% endblock %}
13
14 {% block content %}
15 <!-- NAVIGATION BEGIN -->
16 <!-- TOP NAV BEGIN !!!!!!!!!!!!!!!!!!!!!!! -->
17 <nav class="navbar navbar-inverse navbar-fixed-top">
18 <div class="container-fluid">
19 <div class="navbar-header">
20 <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data
    -target="#navbar" aria-expanded="false" aria-controls="navbar">
21 <span class="sr-only">Toggle navigation</span>
22 <span class="icon-bar"></span>
23 <span class="icon-bar"></span>
24 <span class="icon-bar"></span>
25 </button>
26 <a class="navbar-brand" href="#">FRACS Web</a>
27 </div>
28 <div id="navbar" class="navbar-collapse collapse">
29 <ul class="nav navbar-nav navbar-right">
30 <li><a href="/logout">Log Out</a></li>
31 </ul>
32 </div>
33 </div>
34 </nav>
35 <!-- TOP NAV END !!!!!!!!!!!!!!!!!!!!!!! -->
36
37
38 <!-- SIDE NAV BEGIN !!!!!!!!!!!!!!!!!!!!!!! -->
39 <div class="container-fluid">
40 <div class="row">
41 <div class="col-sm-3 col-md-2 sidebar">
42 <ul class="nav nav-sidebar">
43 <li><a href="/dashboard">Dashboard</a></li>
44 <li><a href="/all_users">FRACS Users</a></li>
45 <li><a href="/view_user_logs">View Access Logs </a></li>
46 </ul>
47 <ul class="nav nav-sidebar">
48 <li class="active" ><a href="/my_profile">User Profile <span class="sr-only">(
    current)</span></a></li>
49 </ul>
50 </div>
51
52 <!-- SIDE NAV END !!!!!!!!!!!!!!!!!!!!!!! -->
53 <div class="col-sm-9 col-sm-offset-3 col-md-10 col-md-offset-2 main">
54 <h2 class="sub-header">User Profile</h2>
55 <div class="col-sm-9 col-sm-offset-3 col-md-10 col-md-offset-2 main">
56 <form>
57 <div class="row">
58 <div class="col-md-2 pr-md-1">
59 <div class="form-group">
60 <label>User ID</label>
61 <input type="text" class="form-control" disabled="" value="{% userprofile.user_id

```

```

    }}">
62 </div>
63 </div>
64 <div class="col-md-3 px-md-1">
65 <div class="form-group">
66 <label>Login Username</label>
67 <input type="text" class="form-control"disabled="" value="{ { userprofile.
    login_username }}">
68 </div>
69 </div>
70 <div class="col-md-7 pl-md-1">
71 <div class="form-group">
72 <label for="exampleInputEmail1">Face ID</label>
73 <input type="email" class="form-control"disabled="" value="{ { userprofile.face_id
    }}">
74 </div>
75 </div>
76 </div>
77 <div class="row">
78 <div class="col-md-6 pr-md-1">
79 <div class="form-group">
80 <label>First Name</label>
81 <input type="text" class="form-control"disabled="" placeholder="Company" value="
    {{userprofile.firstname}}">
82 </div>
83 </div>
84 <div class="col-md-6 pl-md-1">
85 <div class="form-group">
86 <label>Last Name</label>
87 <input type="text" class="form-control"disabled="" placeholder="Last Name" value=
    "{{userprofile.lastname}}">
88 </div>
89 </div>
90 </div>
91 <div class="row">
92 <div class="col-md-12">
93 <div class="form-group">
94 <label>About You</label>
95 <textarea rows="4" cols="80" class="form-control" disabled="" placeholder="Here
    can be your description" value="Mike">An admin/user</textarea>
96 </div>
97 </div>
98 </div>
99 </form>
100
101
102 </div>
103 </div>
104 </div>
105
106 {% endblock %}

```

Listing 50: my_profile.html

```

1  /*
2  * Base structure
3  */
4
5  /* Move down content because we have a fixed navbar that is 50px tall */
6  body {
7  padding-top: 50px;
8  }
9
10
11 /*
12 * Global add-ons
13 */
14
15 .sub-header {
16 padding-bottom: 10px;
17 border-bottom: 1px solid #eee;
18 }
19
20 /*
21 * Top navigation
22 * Hide default border to remove 1px line.
23 */
24 .navbar-fixed-top {
25 border: 0;
26 }
27
28 /*
29 * Sidebar
30 */
31
32 /* Hide for mobile, show later */
33 .sidebar {
34 display: none;
35 }
36 @media (min-width: 768px) {
37 .sidebar {
38 position: fixed;
39 top: 51px;
40 bottom: 0;
41 left: 0;
42 z-index: 1000;
43 display: block;
44 padding: 20px;
45 overflow-x: hidden;
46 overflow-y: auto; /* Scrollable contents if viewport is shorter than content. */
47 background-color: #f5f5f5;
48 border-right: 1px solid #eee;
49 }
50 }
51
52 /* Sidebar navigation */
53 .nav-sidebar {
54 margin-right: -21px; /* 20px padding + 1px border */
55 margin-bottom: 20px;
56 margin-left: -20px;
57 }
58 .nav-sidebar > li > a {
59 padding-right: 20px;
60 padding-left: 20px;
61 }
62 .nav-sidebar > .active > a,
63 .nav-sidebar > .active > a:hover,

```

```

64 .nav-sidebar > .active > a:focus {
65 color: #fff;
66 background-color: #428bca;
67 }
68
69
70 /*
71 * Main content
72 */
73
74 .main {
75 padding: 20px;
76 background-color: #001a33;
77 color: #e6e6ff;
78 }
79 @media (min-width: 768px) {
80 .main {
81 padding-right: 40px;
82 padding-left: 40px;
83 }
84 }
85 .main .page-header {
86 margin-top: 0;
87 }
88
89 tr{
90 border-radius: 30px;
91 border: 2px solid black;
92 color: black;
93 }
94
95 .tbl-headings{
96 color: #e6e6ff;
97 }
98
99 /*
100 * Placeholder dashboard ideas
101 */
102
103 .placeholders {
104 margin-bottom: 30px;
105 text-align: center;
106 }
107 .placeholders h4 {
108 margin-bottom: 0;
109 }
110 .placeholder {
111 margin-bottom: 20px;
112 }
113 .placeholder img {
114 display: inline-block;
115 border-radius: 50%;
116 }

```

Listing 51: all.css

```
1 body {
2 padding-top: 40px;
3 padding-bottom: 40px;
4 background-color: #eee;
5 }
6
7 .form-signin {
8 max-width: 330px;
9 padding: 15px;
10 margin: 0 auto;
11 }
12 .form-signin .form-signin-heading,
13 .form-signin .checkbox {
14 margin-bottom: 10px;
15 }
16 .form-signin .checkbox {
17 font-weight: normal;
18 }
19 .form-signin .form-control {
20 position: relative;
21 height: auto;
22 -webkit-box-sizing: border-box;
23 -moz-box-sizing: border-box;
24 box-sizing: border-box;
25 padding: 10px;
26 font-size: 16px;
27 }
28 .form-signin .form-control:focus {
29 z-index: 2;
30 }
31 .form-signin input[type="email"] {
32 margin-bottom: -1px;
33 border-bottom-right-radius: 0;
34 border-bottom-left-radius: 0;
35 }
36 .form-signin input[type="password"] {
37 margin-bottom: 10px;
38 border-top-left-radius: 0;
39 border-top-right-radius: 0;
40 }
```

Listing 52: all.css