

Python Applications
Advance Python Applications

Signature Assignment Project

Book Management / Library Management Project



Project Supervisor - Prof. Faramarz Mortezaie

Name	Student ID
Nauka Shah	24083814
Aditi Parikh	95991

Table Of Content

Abstract	3
System Environment	3
System Modules	4
Database/ Table Structure	4
Login Module	6
Librarian dashboard	8
Add Book	9
List of Books	10
Delete Book	12
Issue Book	13
Return Book	15
View student logs	16
Face Recognition Module	17
Recognizing a Face with OpenCV HOG Face Detection,Dlib and Face recognition Library	18
Source Code :	19
Conclusion	20
References	22

Abstract

Library management system is a project which aims at developing a computerized system to maintain all the daily work of the library.

The Project is for monitoring and controlling the transactions in a library. It is developed in Python and mainly focuses on basic operations like Adding a new book, editing or deleting a book record. Assign any book to a student and keep track of returns. The project is for the Librarian. He/She can login to the system and perform all the tasks.

This project also includes a module of Artificial Intelligence - Students log entries by their Face recognition. It is useful for the librarian or administrator to sometimes check the records of student entry and exit logs. Instead of manual registration, This project has an automatic approach for this. This AI module captures the faces of students entering the library via camera installed at the library door and then by recognizing their faces from the database, students entry logs are recorded to a separate csv file. Similarly, a camera installed at the door, facing inside the library can keep track of these logs of students exiting the library.

Overall this project of ours is being developed to help the students as well as staff of the library to maintain the library in the best way possible and also reduce the human efforts.

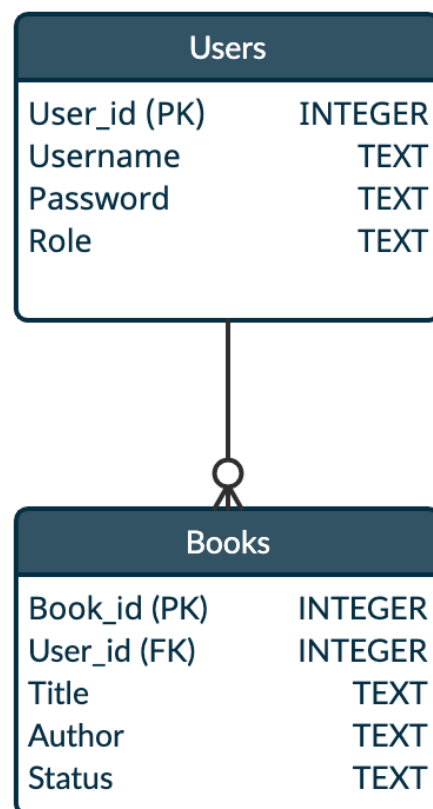
System Environment

- Language - Python
- UI - Tkinter
- Libraries Used - OS, OpenCV, Dlib, Face_Recognition, NumPy, CSV, DateTime
- Database - Sqlite3

System Modules

- Login Module of Librarian
- Book management Module
- Assign/Release a book module
- Face recognition module

Database/ Table Structure



```
conn = None
conn = sqlite3.connect('library_info.db')
conn.execute("PRAGMA foreign_keys = 1")
cur = conn.cursor()
try:
    cur.execute('''CREATE TABLE IF NOT EXISTS Users(User_id INTEGER PRIMARY KEY NOT NULL,
                                                    Username TEXT UNIQUE,
                                                    Password TEXT,
                                                    Role TEXT)''')
    conn.commit()

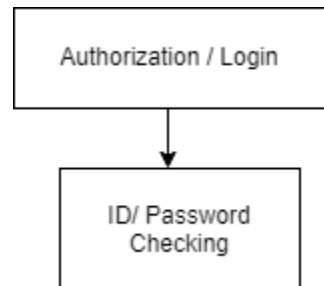
except sqlite3.Error as err:
    print('Database Error ', err)

print("creating another table..")
cur.execute('''CREATE TABLE IF NOT EXISTS Books(Book_id INTEGER PRIMARY KEY NOT NULL, Title TEXT,
                                                Author TEXT,
                                                Status TEXT,
                                                User_id INTEGER NOT NULL,
                                                FOREIGN KEY (User_id) REFERENCES Users (User_id)
                                                ON DELETE CASCADE
                                                ON UPDATE CASCADE)''')
```

Login Module

Role Type: Librarian

Authorization: Login with provided User ID and Password



Validate username and password, if record matches to the database user will be logged in to the system.

```
def validate():
    print("in validate function 2")
    name = entry1.get()
    password = entry2.get()

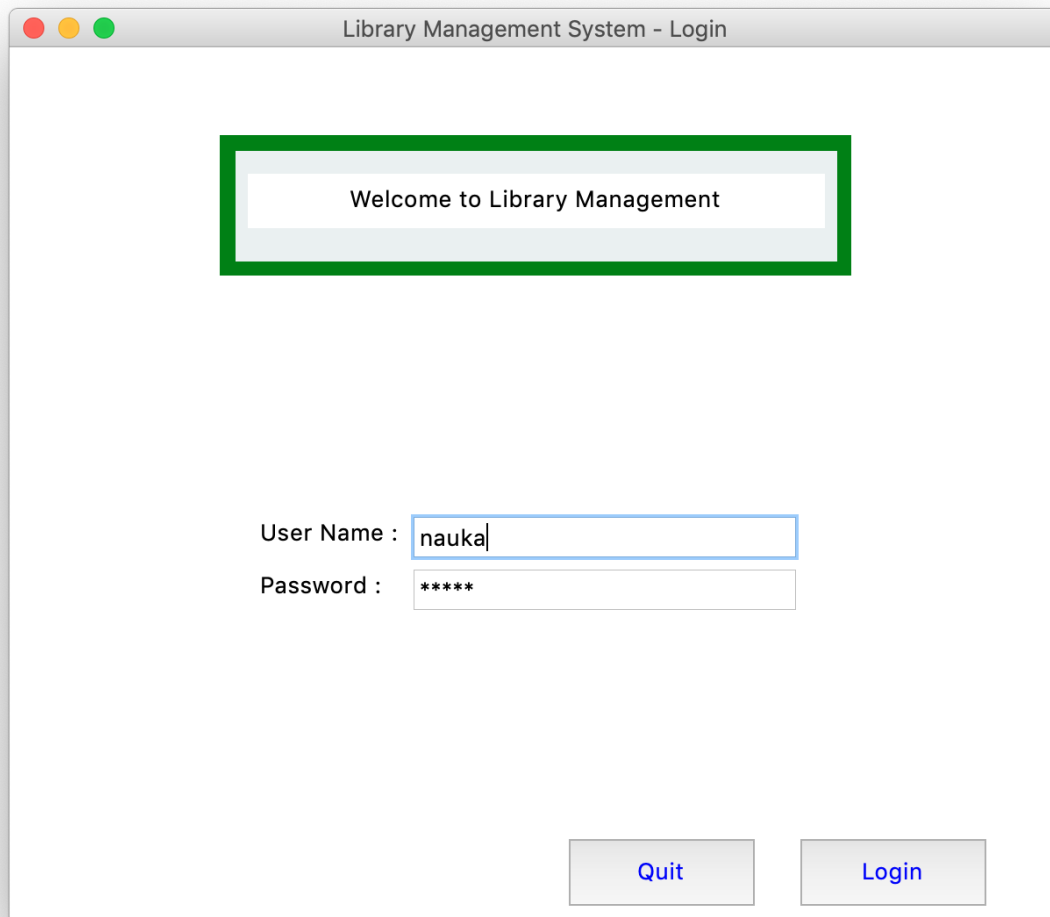
    cur.execute('''SELECT * FROM Users WHERE Users.Username = ? AND Users.Password = ?''', (name, password))
    results = cur.fetchall()

    if len(results) == 1:
        print("validation matches and perform book operations")
        cur.execute('''SELECT User_id FROM Users WHERE Users.Username = ? AND Users.Password = ?''',
                    (name, password))
        result = cur.fetchall()

        global logged_in_id
        issueBook.logged_in_id = result
        returnBook.logged_in_id = result

        manageBookOperations()
    else:
        messagebox.showerror("Error", "Invalid Credentials,Please try again!")
```

Screenshot:



Library Management System - Login

Welcome to Library Management

User Name : nauka

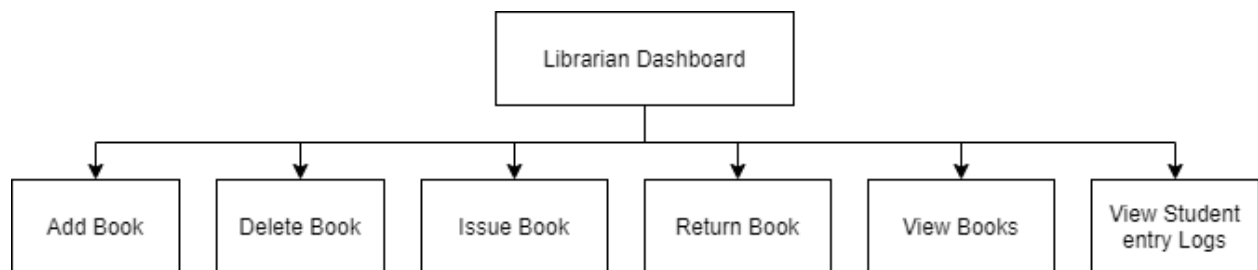
Password : *****

Quit Login

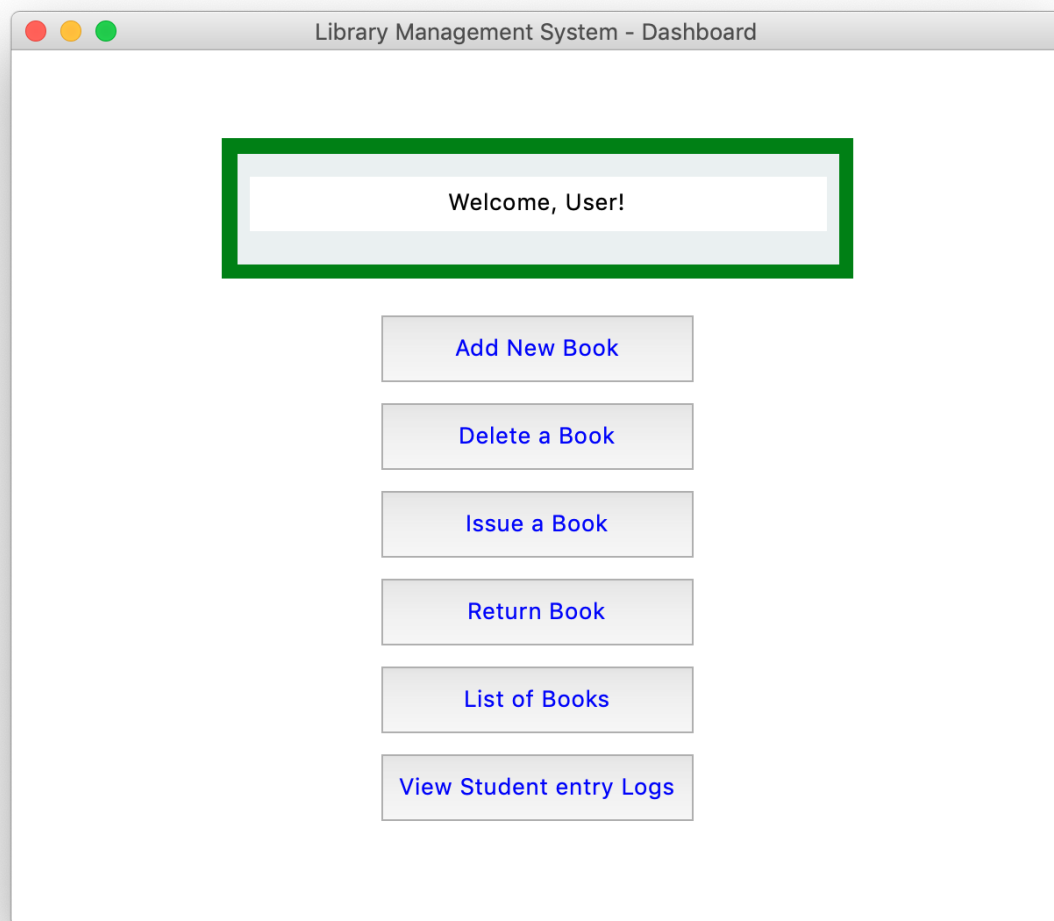
Librarian Dashboard - Book Management Module

Once Librarian logs in with the correct credentials, He/She will be redirected to the dashboard screen. Where he/she can perform following tasks:

- 1) Adding a new Book into the database
- 2) Deleting any book record from the database
- 3) Issue a book to the student
- 4) Manage return of a book from student and book's availability
- 5) View the entire list of books
- 6) View the logs of Student entries.



Librarian dashboard



Add Book

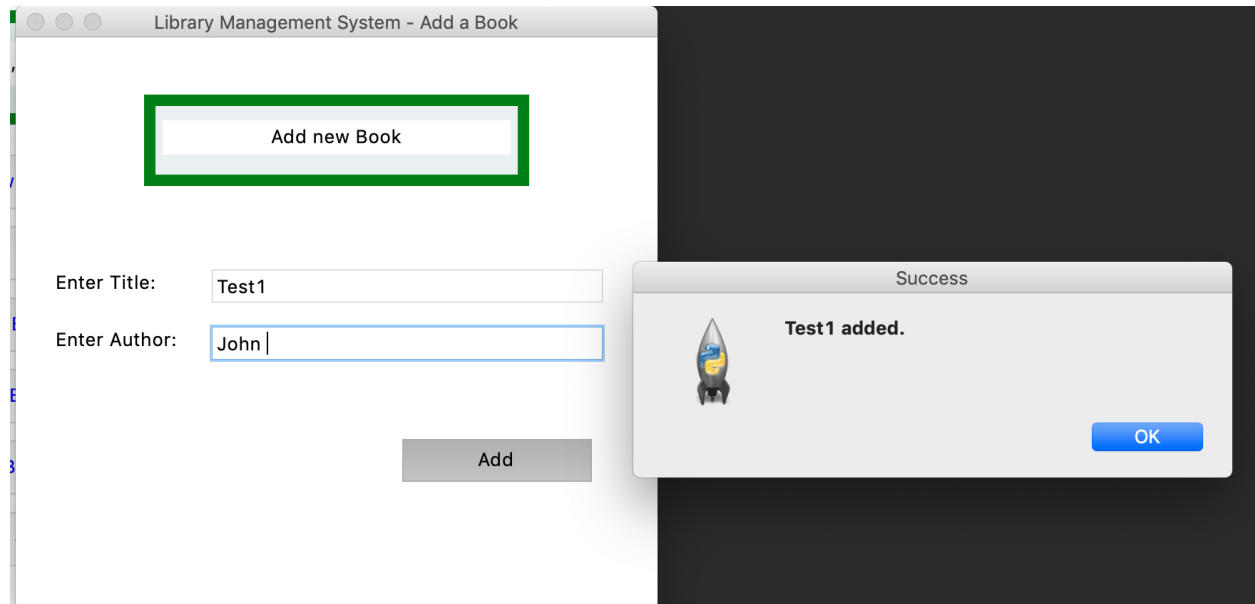
Adding a new book in the database and if the title is empty it will prompt the user to enter book title and author name. Added books will be by default added to the id of librarian.

```
try:
    if btitle == '' or bauthor == '':
        messagebox.showinfo('Warning', "Input required")
    else:
        cursor.execute('''INSERT INTO Books(Title, Author, Status, User_id )
                        VALUES(?, ?, ?, ?)''', (btitle, bauthor, "available", 1))

        conn.commit()
        messagebox.showinfo('Success', btitle+" added.")
        window.destroy()

except:
    messagebox.showerror("Error", "Could not add given book data into Database!")
```

Screenshot :



List of Books

View list of books and display in the GUI.

```
try:
    conn = None
    conn = sqlite3.connect('library_info.db')
    cur = conn.cursor()
    cur.execute('SELECT * FROM Books')
    results = cur.fetchall()
    total_rows = len(results)
    total_columns = len(results[0])

    lst = [('ID', 'Title', 'Author', 'Status', 'Assigned To')]
    for k in range(0, 5):
        e = Entry(tableFrame1, width=10, fg='black')
        e.grid(row=0, column=k)
        e.insert(END, lst[0][k])

    for i in range(total_rows):
        for j in range(total_columns):
            e = Entry(tableFrame1, width=10, fg='black')

            e.grid(row=i + 1, column=j)
            e.insert(END, results[i][j])

except:
    messagebox.showerror("Error", "Cannot Fetch data.")

finally:
    if conn != None:
        conn.close()
```

Screenshot :

Library Management System - List of Books

List of Books

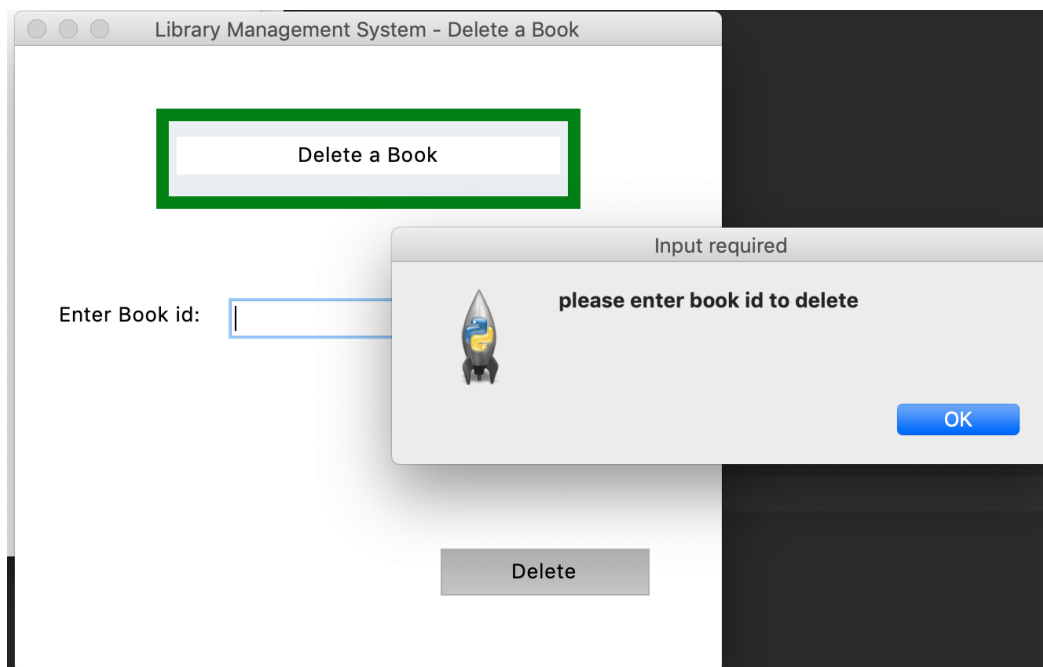
ID	Title	Author	Status	Assigned To
1	book1	author1	available	1
2	test2	author2	no	1
3	test	a1	available	1
5	Test1	John	available	1

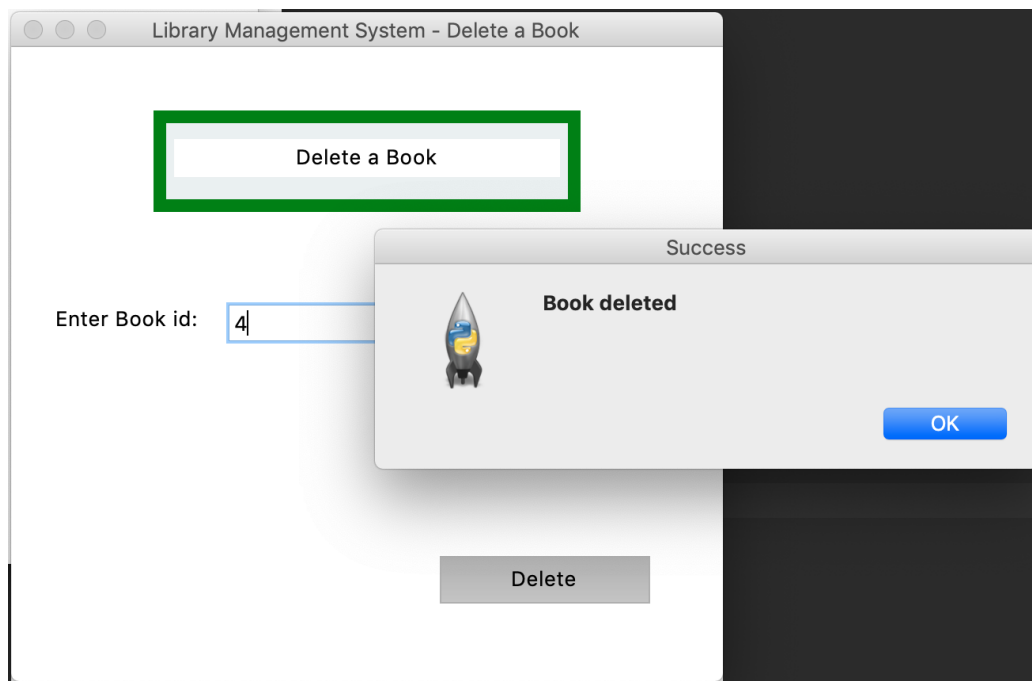
Close

Delete Book

Books with the entered id will be removed from the database if found. Users cannot submit empty values to remove records.

```
def delete_db():  
    global id  
  
    bid = id.get()  
  
    print(bid, end='--')  
    print("delete")  
  
    try:  
        print(bid)  
        if bid == '':  
            messagebox.showinfo('Input required', "please enter book id to delete")  
        else:  
            cursor.execute('DELETE FROM Books WHERE Book_id = ?', (bid,))  
            conn.commit()  
            messagebox.showinfo('Success', "Book deleted")  
            window.destroy()  
  
    except:  
        messagebox.showerror("Error", "Book with given id does not exist!")
```





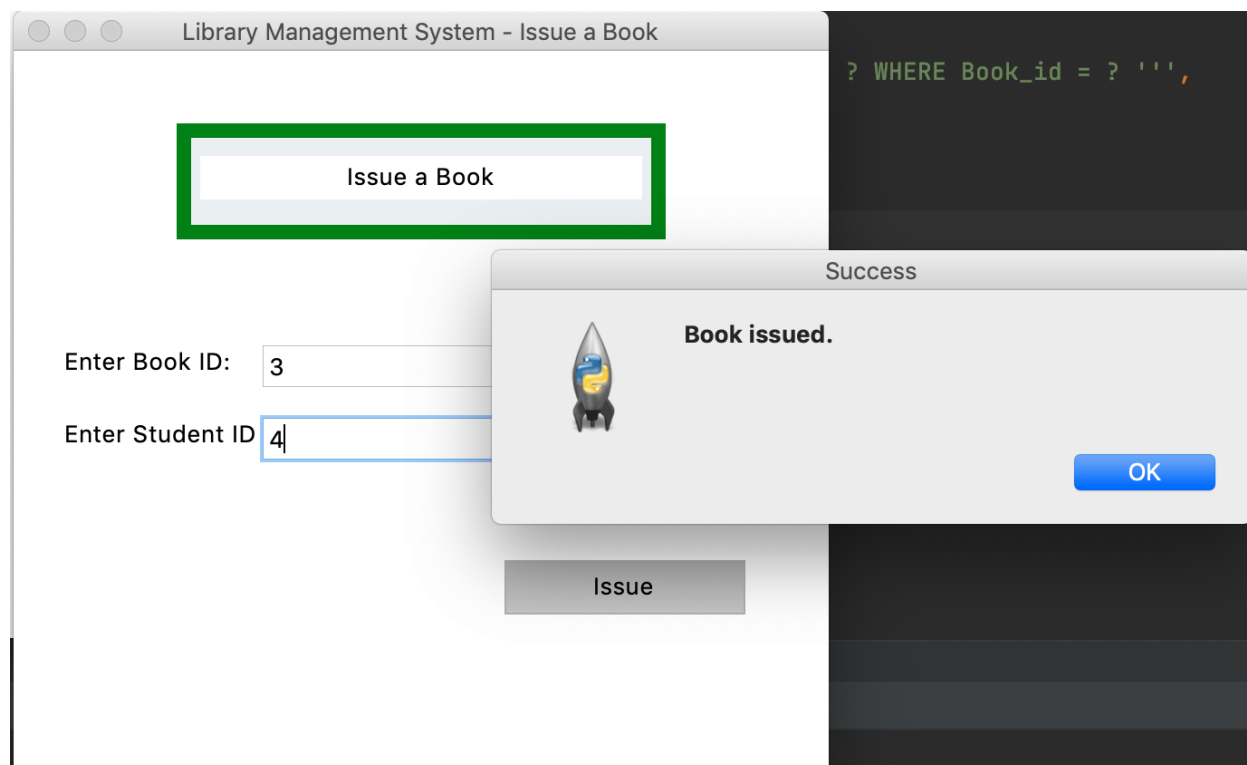
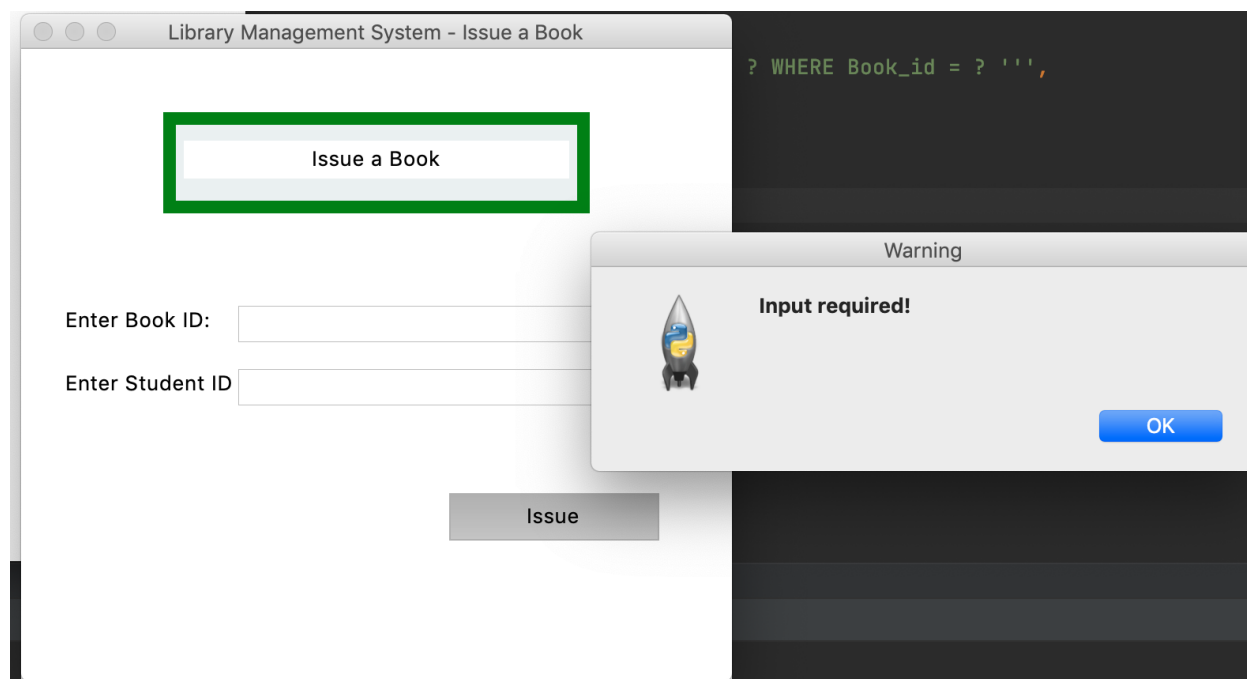
Issue Book

Book will be issued for the entered user id if the status is available. User cannot input empty values to issue books.

```
try:
    # get list of all available book, if the one user want to issue
    # is available, user can (update) issue to userid.
    cursor.execute('SELECT Book_id from Books where Status = 'Available' or Status = 'available' ')
    result = cursor.fetchall()
    if bid == '':
        messagebox.showinfo('Warning', "Input required!")

    for i in result:
        if int(i[0]) == int(bid):
            print("to change value of flag check each i", i[0], bid)
            cursor.execute('UPDATE Books SET Status = ?, User_id = ? WHERE Book_id = ? ',
                           ("no", int(bStudentId), int(bid)))
            conn.commit()
            messagebox.showinfo('Success', "Book issued.")
            window.destroy()
            break
        else:
            print("Error", "Required Book is not available!")
except:
    messagebox.showerror("Error", "Cannot issue given book!")
```

Screenshots:

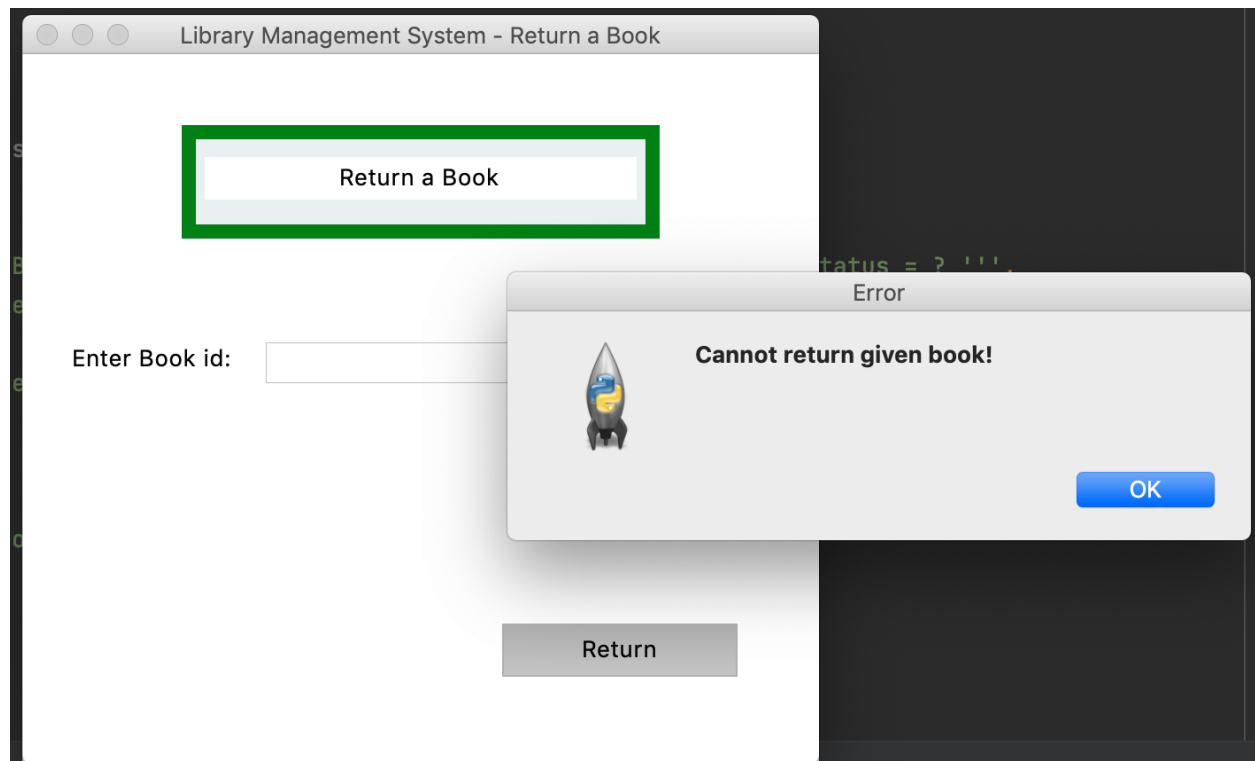


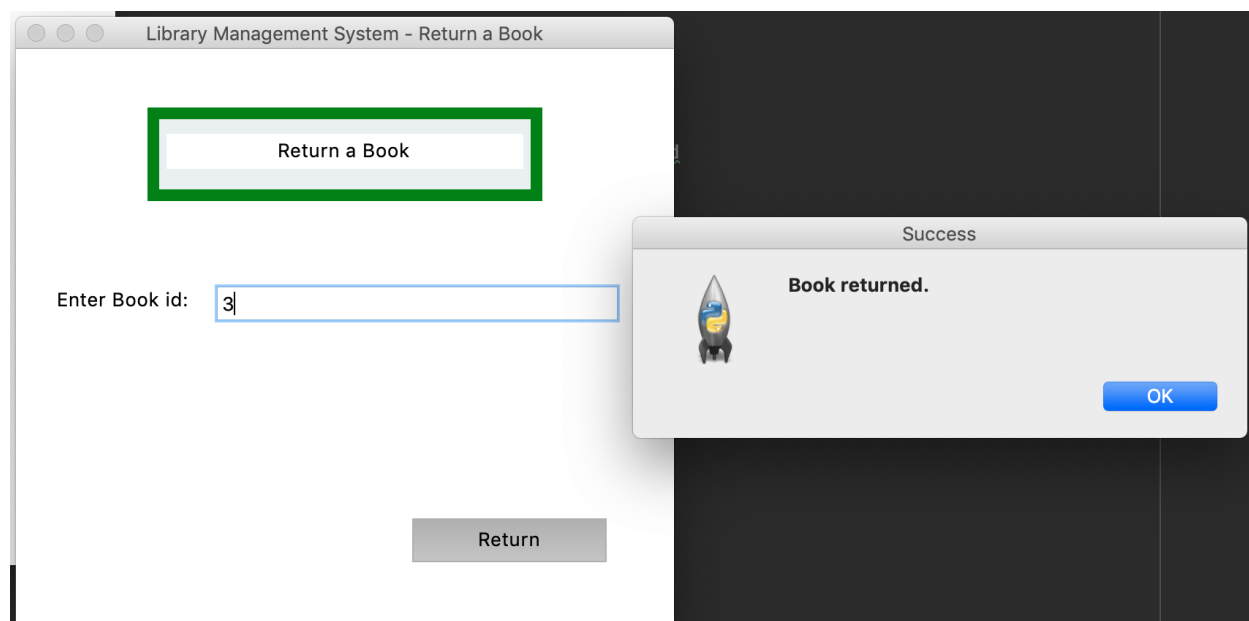
Return Book

Status for the entered book id will be updated to Available if book id is found from the database. Cannot submit empty value to initiate return.

```
try:
    # update entered book id's status as available and set user id = tempid
    for i in logged_in_id:
        temp_id = i[0]
        cursor.execute('''UPDATE Books SET Status = ?, User_id = ? WHERE Book_id = ? and Status = ? ''',
                        ("available", int(temp_id), int(bid), "no"))
        conn.commit()
        messagebox.showinfo('Success', "Book returned.")
        window.destroy()

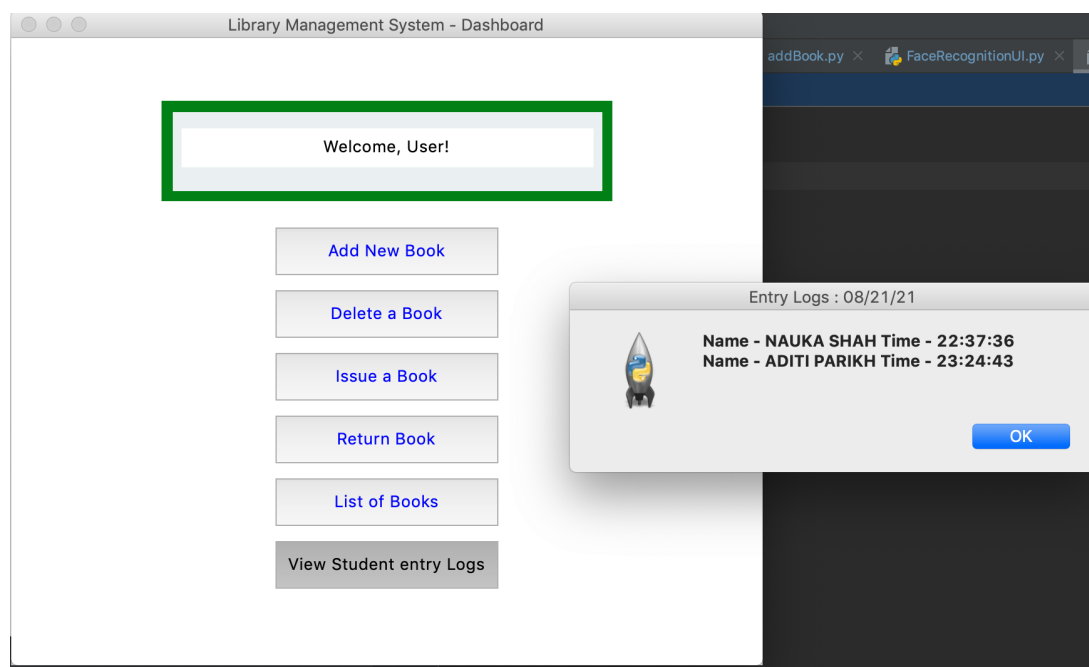
except:
    messagebox.showerror("Error", "Cannot return given book!")
```





View student logs

Librarian can view student entry logs fetched from face recognition module to track the students who has entered in the library to manage attendance/entries.

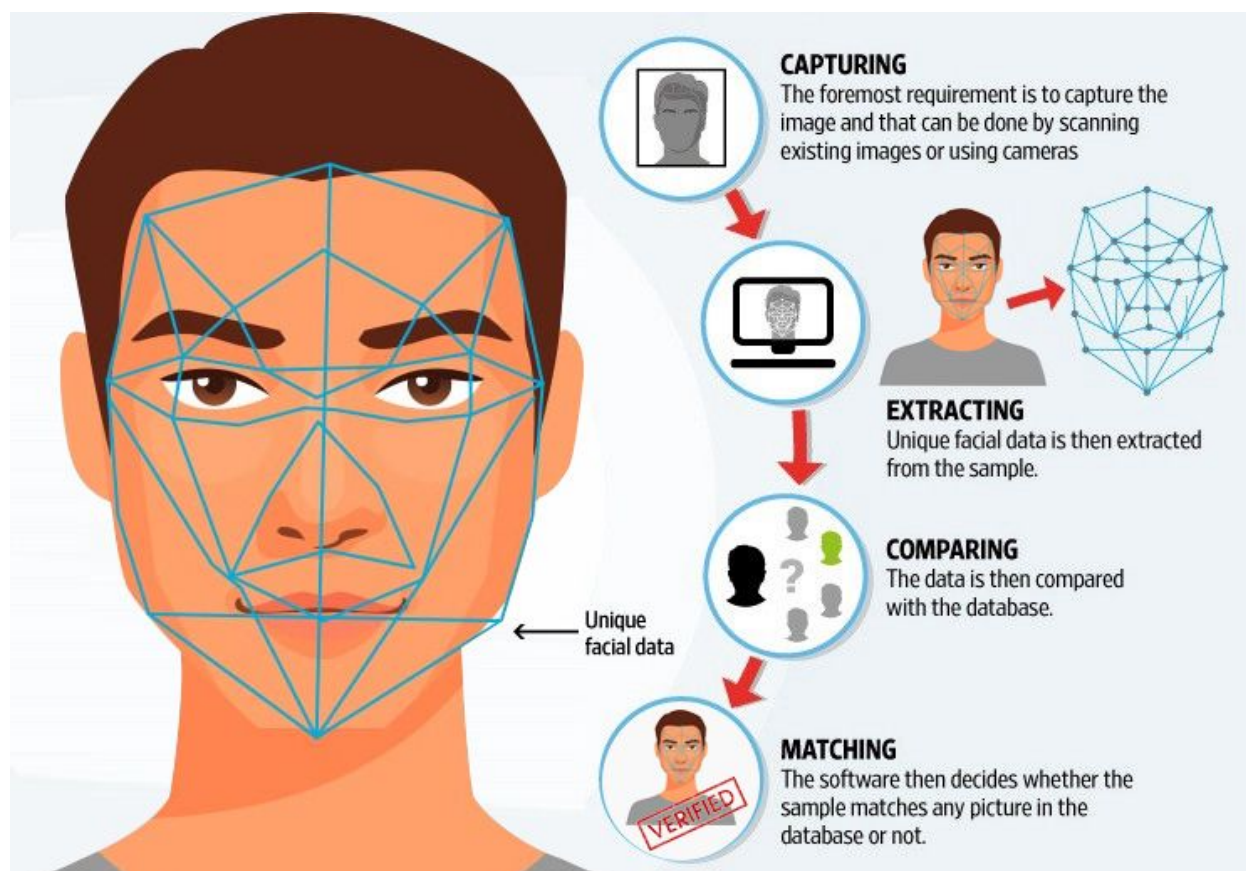


Face Recognition Module

Face recognition extends beyond detecting the presence of a human face to determine whose face it is.

After the detection of the face is done, we extract specific features from it. A neural network takes an image of the face of the person as input and outputs a vector that represents the most important features of a face! In machine learning, this vector is called embedding and hence we call this vector face embedding.

When we train the neural network, the network learns to output similar vectors for faces that look similar.



Recognizing a Face with OpenCV HOG Face Detection,Dlib and Face recognition Library

- 1) Encode a picture using the HOG algorithm to create a simplified version of the image. Using this simplified image, find the part of the image that most looks like a generic HOG encoding of a face.
- 2) Figure out the pose of the face by finding the main landmarks in the face. Once we find those landmarks, use them to warp the image so that the eyes and mouth are centered.
- 3) Pass the centered face image through a neural network that knows how to measure features of the face. Save those 128 measurements.
- 4) Looking at all the faces we've measured in the past, see which person has the closest measurements to our face's measurements. That's our match!

```
import cv2
import face_recognition

imgElon =
face_recognition.load_image_file('Face_Recognition/Images/Elon-Musk.jpg')
imgElon = cv2.cvtColor(imgElon,cv2.COLOR_BGR2RGB)
imgTest =
face_recognition.load_image_file('Face_Recognition/Images/Elon-Musk.jpg')
imgTest = cv2.cvtColor(imgTest,cv2.COLOR_BGR2RGB)

faceLoc = face_recognition.face_locations(imgElon)[0]
encodeElon = face_recognition.face_encodings(imgElon)[0]
print(encodeElon)
cv2.rectangle(imgElon, (faceLoc[3],faceLoc[0]), (faceLoc[1],faceLoc[2]), (255
,0,255),2) # top, right, bottom, left

faceLocTest = face_recognition.face_locations(imgTest)[0]
encodeTest = face_recognition.face_encodings(imgTest)[0]
cv2.rectangle(imgTest, (faceLocTest[3],faceLocTest[0]), (faceLocTest[1],face
LocTest[2]), (255,0,255),2)

results = face_recognition.compare_faces([encodeElon], encodeTest) # True
or False.based on test img is same as given.
```

```
faceDis = face_recognition.face_distance([encodeElon], encodeTest) # Lower
the distance, better the match.
output = ''
print('Face Diffrence :',{round(faceDis[0],2)})
if str(results[0]) == 'True':
    output = 'Matches'
else:
    output = 'Does not match'

cv2.putText(imgTest,f'{output}',(50,50),cv2.FONT_HERSHEY_COMPLEX,1,(255,0,
255),3)

cv2.imshow('Original Image',imgElon)
cv2.imshow('Test Image',imgTest)
cv2.waitKey(0)
```

Source Code

Please find the code for this project from this repository:

<https://github.com/itu-edu/library-management>

Conclusion

To conclude, This report contains the idea, introduction and details of our signature course project - Library Management system. The system is implemented in Python language with sqlite Database integration.

We have developed this system and tried to accommodate all the major concepts like Tkinter UI, Database functions, Global variables and their usage, Validations of fields and catching exceptions. Apart from these basic concepts, we researched for an advanced concept of AI - Face Recognition and integrated this module in our project. For this we have used OpenCV library and Face-Recognition library.

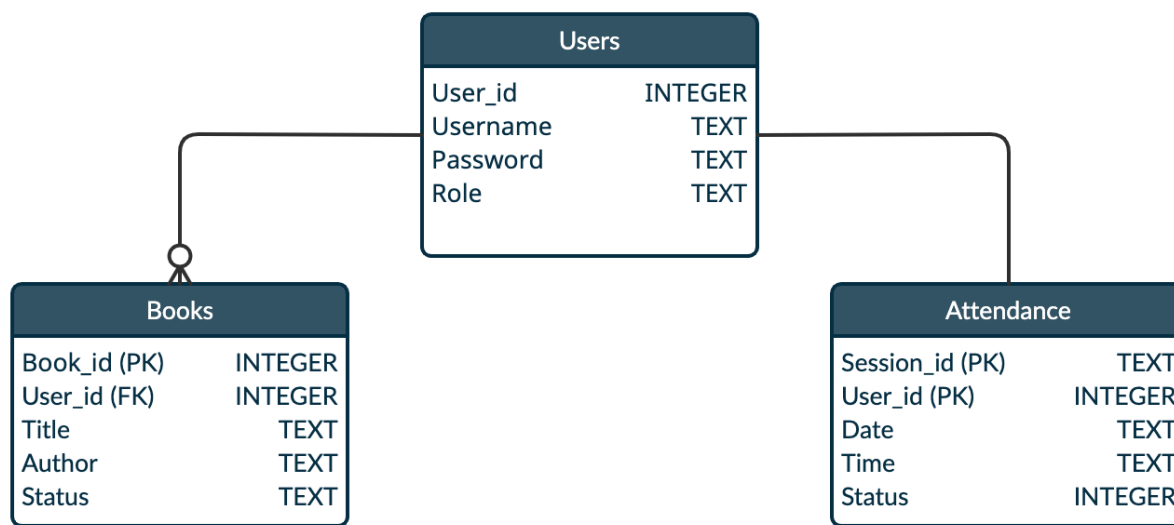
The purpose of this project was to implement all the concepts we have learned during the course. In the process of developing the software, We have learned how to actually implement all concepts which are linked together to achieve one specific goal.

Learning Outcome:

- More detailed knowledge about the concepts
- Ability to solve the technical issues
- Learned to work in a team
- Project management and Collaboration

Future Enhancement:

Attendance With Face recognition module can be expanded in future. We are thinking of extending it and Creating a whole new table of attendance which will keep track of attendance of students. In our project, we have considered only entry logs of students entering the library, but in future, similarly We can also implement the feature of capturing the students who are exiting the library in the module.



Also, In the future, We are thinking of extending the system and have Role based login. We will expand this system's scope to have two roles - Librarian and Student. All features available in the dashboard will be based on the role.

References

1. <https://realpython.com/face-recognition-with-python/>
2. <https://towardsdatascience.com/a-guide-to-face-detection-in-python-3eab0f6b9fc1>
3. <https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e>
4. <https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cfc121d78>