## Certificate

This is to certify that Mustafa Hasan (TL), Rajat Kaliya, Kartik Sharma, Kapil Dev Sharma, Ashu Hasan has satisfactorily completed the project work entitled **“Data Augmentation”** And Prepared this project during the academic year 2020-2021.In partial fulfillment for the award of ConsoleLancer, Bangalore. It is further certified that they completed all required phases of the project.

Project Guide Managing Director

## Acknowledgement

We articulate our sincere gratitude to all those who helped us in making this venture a grand success, without whose constructive criticism as well as words of inspiration this project of ours would not have seen the light.

We take this opportunity to thank Mr. SHOUHADDO PAUL, for the knowledge and guidance provided to us on the project work. We gratefully thank them for extending to us their invaluable time and resources.

We would also like to mention our sincere gratitude to Mr.SHUBHAM TANDON, Managing Director, ConsoleLancer, for giving us opportunity to work in this project at ConsoleLancer

Mustafa Hasan (TL),

Rajat Kaliya, Kartik Sharma,

Kapil Dev Sharma,

Ashu Hasan

## INDEX

* Data augmentation
* Transformation of images
* Tkinter
* Tkinter modules
* The packer
* Tk option data types
* OpenCV
* OpenCV python
* NumPy
* My sql
* Data Base and Table
* Filters
* Gant Chart
* Pert Chart
* ULM Diagram
* ER Diagram
* Flow Diagram
* Screenshots
* Coding

**Data augmentation**

**Data augmentation** in data analysis are techniques used to increase the amount of data by adding slightly modified copies of already existing data or newly created synthetic data from existing data. It acts as a regularizer and helps reduce overfitting when training a machine learning model. It is closely related to oversampling in data analysis.

**Data augmentation** is a strategy that enables practitioners to significantly increase the diversity of **data** available for training models, without actually collecting new **data**. **Data augmentation** techniques such as cropping, padding, and horizontal flipping are commonly used to train large neural networks.

**Data augmentation** can be used to increased the accuracy and learning of the model because by using Data augmentation we can create several images of same image in different perspective like by flipping, rotating, sharpening and many more so that same image can be train in multiple ways and model can easily predict the image by so much learning .

**Transformation of images**

* Geometric transformations,
* Flipping,
* Color modification,
* Cropping,
* Rotation,
* Noise injection
* Random erasing are used to augment image in deep

learning.

### Tkinter

The tkinter package (“Tk interface”) is the standard Python interface to the Tk GUI toolkit. Both Tk and tkinter are available on most Unix platforms, as well as on Windows systems. (Tk itself is not part of Python; it is maintained at ActiveState.) Running python -m tkinter from the command line should open a window demonstrating a simple Tk interface, letting you know that tkinter is properly installed on your system, and also showing what version of Tcl/Tk is installed, so you can read the Tcl/Tk documentation specific to that version.

#### Tkinter modules

Tkinter Modules Most of the time, tkinter is all you really need, but a number of additional modules are available as well. The Tk interface is located in a binary module named tkinter. This module contains the lowlevel interface to Tk, and should never be used directly by application programmers. It usually a shared library (or DLL), but might in some cases be statically linked with the Python interpreter. In addition to the Tk interface module, tkinter includes a number of Python modules, tkinter.constants being one of the most important. Importing tkinter will automatically import tkinter.constants, so, usually, to use Tkinter all you need is a simple import statement

**The Packer**

The packer is one of Tk’s geometry-management mechanisms. Geometry managers are used to specify the relative positioning of widgets within their container - their mutual master. In contrast to the more cumbersome placer (which is used less commonly, and we do not cover here), the packer takes qualitative relationship specification - above, to the left of, filling, etc - and works everything out to determine the exact placement coordinates for you. The size of any master widget is determined by the size of the “slave widgets” inside. The packer is used to control where slave widgets appear inside the master into which they are packed. You can pack widgets into frames, and frames into other frames, in order to achieve the kind of layout you desire. Additionally, the arrangement is dynamically adjusted to accommodate incremental changes to the configuration, once it is packed. Note that widgets do not appear until they have had their geometry specified with a geometry manager. It’s a common early mistake to leave out the geometry specification, and then be surprised when the widget is created but nothing appears. A widget will appear only after it has had, for example, the packer’s pack() method applied to it. The pack() method can be called with keyword-option/value pairs that control where the widget is to appear within its container, and how it is to behave when the main application window is resized.

### Tk Option Data Types

*Anchor*

Legal values are points of the compass: "n", "ne", "e", "se", "s", "sw", "w",

"nw", and also "center

*Bitmap*

There are eight built-in, named bitmaps: 'error', 'gray25', 'gray50', 'hourglass', 'info', 'questhead', 'question', 'warning'. To specify an X bitmap filename, give the full path to the file, preceded with an @, as in "@/usr/contrib/bitmap/gumby.bit".

*Boolean*

You can pass integers 0 or 1 or the strings "yes" or "no".

*callback*

This is any Python function that takes no arguments

*Color*

Colors can be given as the names of X colors in the rgb.txt file, or as strings

representing RGB values in 4 bit: "#RGB", 8 bit: "#RRGGBB", 12 bit” "#RRRGGGBBB", or 16 bit "#RRRRGGGGBBBB" ranges, where R,G,B

here represent any legal hex digit. See page 160 of Ousterhout’s book for details.

*cursor*

The standard X cursor names from cursorfont.h can be used, without the XC prefix. For example to get a hand cursor (XC hand2), use the string "hand2". You can also specify a bitmap and mask file of your own. See page 179 of

Ousterhout’s book.

*Distance*

Screen distances can be specified in either pixels or absolute distances. Pixels are given as numbers and absolute distances as strings, with the trailing character denoting 9 units: c for centimetres, i for inches, m for millimetres, p for printer’s points. For example, 3.5 inches is expressed as "3.5i"

*font*

Tk uses a list font name format, such as {courier 10 bold}. Font sizes with positive numbers are measured in points; sizes with negative numbers are measured in pixels.

*Geometry*

This is a string of the form widthxheight, where width and height are measured in pixels for most widgets (in characters for widgets displaying text). For example: fred["geometry"] = "200x100".

*Justify*

Legal values are the strings: "left", "center", "right", and "fill".

*Region*

This is a string with four space-delimited elements, each of which is a legal distance (see above). For example: "2 3 4 5" and "3i 2i 4.5i 2i" and "3c 2c 4c 10.43c" are all legal regions.

*Relief*

Determines what the border style of a widget will be. Legal values are: "raised", "sunken", "flat", "groove", and "ridge".

*Scrollcommand*

This is almost always the set() method of some scrollbar widget, but can be any widget method that takes a single argument.

*Wrap*

Must be one of: "none", "char", or "word".

### OPENCV

OpenCV was started at Intel in 1999 by **Gary Bradsky**, and the first release came out in 2000. **Vadim Pisarevsky** joined Gary Bradsky to manage Intel's Russian software OpenCV team. In 2005, OpenCV was used on Stanley, the vehicle that won the 2005 DARPA Grand Challenge. Later, its active development continued under the support of Willow Garage with Gary Bradsky and Vadim Pisarevsky leading the project. OpenCV now supports a multitude of algorithms related to Computer Vision and Machine Learning and is expanding day by day.

OpenCV supports a wide variety of programming languages such as C++,

Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android, and iOS. Interfaces for high-speed GPU operations based on CUDA and OpenCL are also under active development.

OpenCV-Python is the Python API for OpenCV, combining the best qualities of the OpenCV C++ API and the Python language

##### OPENCV PYTHON

OpenCV-Python is a library of Python bindings designed to solve computer vision problems.

Python is a general purpose programming language started by **Guido van Rossum** that became very popular very quickly, mainly because of its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing readability.

Compared to languages like C/C++, Python is slower. That said, Python can be easily extended with C/C++, which allows us to write computationally intensive code in C/C++ and create Python wrappers that can be used as Python modules. This gives us two advantages: first, the code is as fast as the original C/C++ code (since it is the actual C++ code working in background) and second, it easier to code in Python than C/C++. OpenCV-Python is a Python wrapper for the original OpenCV C++ implementation.

OpenCV-Python makes use of **Numpy**, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

### NumPy

NumPy is a general-purpose array-processing package. It provides a highperformance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

A powerful N-dimensional array object

Sophisticated (broadcasting) functions

Tools for integrating C/C++ and Fortran code

Useful linear algebra, Fourier transform, and random number capabilities Besides its obvious scientific uses, NumPy can also be used as an efficient multidimensional container of generic data.

Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

### My sql

[MySQL is](https://www.mysql.com/) one of the most popular [database management systems (DBMSs) on](https://en.wikipedia.org/wiki/Database#Database_management_system) the market today. It ranked second only to the [Oracle DBMS in](https://docs.oracle.com/cd/E11882_01/server.112/e40540/intro.htm) this year’s [DBEngines Ranking. A](https://db-engines.com/en/ranking)s most software applications need to interact with data in some form, programming languages like Python provide tools for storing and accessing these data sources.

Using the techniques discussed in this tutorial, you’ll be able to efficiently integrate a MySQL database with a Python application. You’ll develop a small MySQL database for a movie rating system and learn how to query it directly from your Python code.

* Identify unique features of **MySQL**
* **Connect your application** to a MySQL database
* Query the database to **fetch required data**
* **Handle exceptions** that occur while accessing the database • Use **best practices** while building database applications

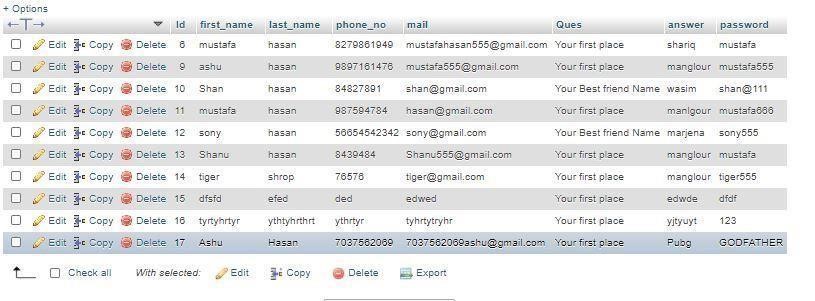
To get the most out of this tutorial, you should have a working knowledge of

Python concepts like [for loop](https://realpython.com/python-for-loop/)[s, function](https://realpython.com/defining-your-own-python-function/)[s, exception handling, a](https://realpython.com/python-exceptions/)nd installing Python packages using [pip. Y](https://realpython.com/what-is-pip/)ou should also have a basic understanding of relational database management systems and SQL queries like SELECT, DROP, CREATE, and JOIN.

## Data Base



**User Table**

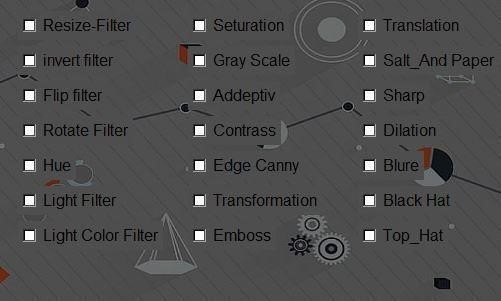


### Admin Table

### Filters

we are using so many filters in this software these are namely given below:-

* Resize Image
* Crop Image
* Padding Image
* Flip Image
* Superpixel Image
* Segment Colorfulness
* Invert Image
* Add Light
* Add Light Color
* Saturation Image
* Hue Image
* Multiply Image
* Gausian Blur
* Averageing Blur
* Median Blur
* Bileteralblur
* Erosion Image
* Dilation Image
* Opening Image
* Closing Image
* Morphological Gradient Image
* Top Hat Image
* Black Hat Image
* Sharpen Image
* Emboss Image
* Edge Image
* Adaptive Gaussian Noise
* Salt Image
* Paper Image
* Salt and Paper Image
* Contrast Image
* Edge Detect Canny Image
* Grayscale Image
* Scale Image
* Translation Image
* Rotate Image
* Transformation Image



**Note: -** You can select to any one or all filter to generate your sample for data augmentation at once.

All filter has rand function to generate random samples.

**Gant Chart**

Researching

10 November 2020

Documentation

20 February

Implementation & Testing

23 January

Coding

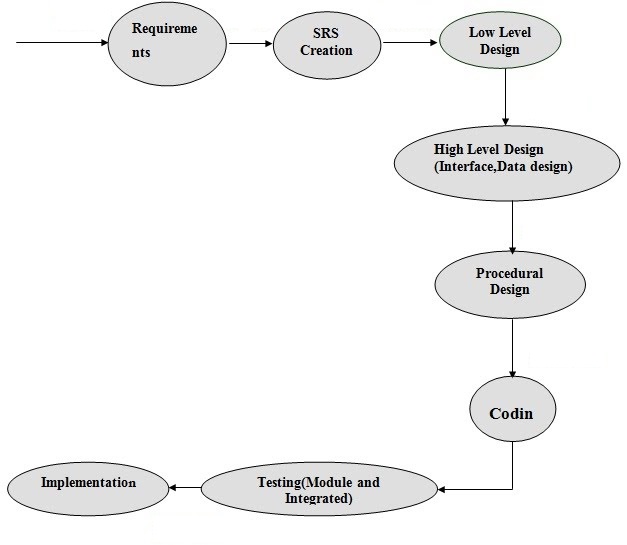
7 January

Designing

7 January

Planning

21 December

**PERT Chart**

**UML Diagrams:**

**Actor:**  
 A coherent set of roles that users of use cases play when interacting with the use `cases.

**Use case:**

A description of sequence of actions, including variants, that a system performs that yields an observable result of value of an actor.

**USECASE DIAGRAM:**

A Use case is a description of set of sequence of actions. Graphically it is rendered as an ellipse with solid line including only its name. Use case diagram is a behavioral diagram that shows a set of use cases and actors and their relationship. It is an association between the use cases and actors. An actor represents a real-world object. Primary Actor – Sender, Secondary Actor Receiver.

**User Use case**

**Admin Use case**

The Entity-Relationship (ER) model was originally proposed by Peter in 1976 [Chen76] as a way to unify the network and relational database views. Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represents data objects. Since Chen wrote his paper the model has been extended and today it is commonly used for database design For the database designer, the utility of the ER model is:

* it maps well to the relational model. The constructs used in the ER model can easily be transformed into relational tables.
* it is simple and easy to understand with a minimum of training. Therefore, the model can be used by the database designer to communicate the design to the end user.
* In addition, the model can be used as a design plan by the database developer to implement a data model in a specific database management software.

**Connectivity and Cardinality**

The basic types of connectivity for relations are: one-to-one, one-to-many, and many-to-many. A *one-to-one* (1:1) relationship is when at most one instance of a entity A is associated with one instance of entity B. For example, "employees in the company are each assigned their own office. For each employee there exists a unique office and for each office there exists a unique employee.

A *one-to-many* (1:N) relationships is when for one instance of entity A, there are zero, one, or many instances of entity B, but for one instance of entity B, there is only one instance of entity A. An example of a 1:N relationships is a department has many employees each employee is assigned to one department

A *many-to-many* (M:N) relationship, sometimes called non-specific, is when for one instance of entity A, there are zero, one, or many instances of entity B and for one instance of entity B there are zero, one, or many instances of entity A. The connectivity of a relationship describes the mapping of associated

**ER Notation**

There is no standard for representing data objects in ER diagrams. Each modeling methodology uses its own notation. The original notation used by Chen is widely used in academics texts and journals but rarely seen in either CASE tools or publications by non-academics. Today, there are a number of notations used, among the more common are Bachman, crow's foot, and IDEFIX.

All notational styles represent entities as rectangular boxes and relationships as lines connecting boxes. Each style uses a special set of symbols to represent the cardinality of a connection. The notation used in this document is from Martin. The symbols used for the basic ER constructs are:

* **Entities** are represented by labeled rectangles. The label is the name of the entity. Entity names should be singular nouns.
* **Relationships** are represented by a solid line connecting two entities. The name of the relationship is written above the line. Relationship names should be verbs
* **Attributes**, when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.
* **Cardinality** of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one.
* **Existence** is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional

**ER Diagram**

Admin

User

Sign In

Sign Up

User

Download Information

Admin

**Flow Diagram**

These Diagram will help to identify the flow of the project by some figures.

Each shape have there own meaning or user for specific purpose which will help you to better understanding for the flow.

Such as:

1. **Window**
2. **Button**
3. **Fields**
4. **Check Button**
5. **Relation Indicator Line**

### USER PHASE

### 

### 

### 1.WINDOW

ID Or Number

Sign In

User Sign In

Sign Up

Password

Read Me



2.WINDOW

Confirm Password

Sign In

User Sign Up

Sign Up

Password

Read Me

Answer



Security Question

E-mail ID

Phone Number

Last Name

First Name

##### 3.WINDOW

User Dash Board

Show Feature

Upload

Select Path



Hide Features

Download

Hide Filter

Select Filter

##### 4.WINDOW

Sharp Dilation Blur Top Hat Black Hat

Resize Translation

Invert Emboss

Flip  Transformation

USER DASHDOARD

Rotate Edge

Hue Contrast

Light Addeptive

Colour Light

Seturation GrayScale Salt And Paper

##### 5.WINDOW

ID

Sign In

Admin Sign In

Password

Read Me



##### 6.WINDOW

Show User

Admin Dashboard

User Name

Sign Out

User Number

Download

User ID



##### 7.WINDOW

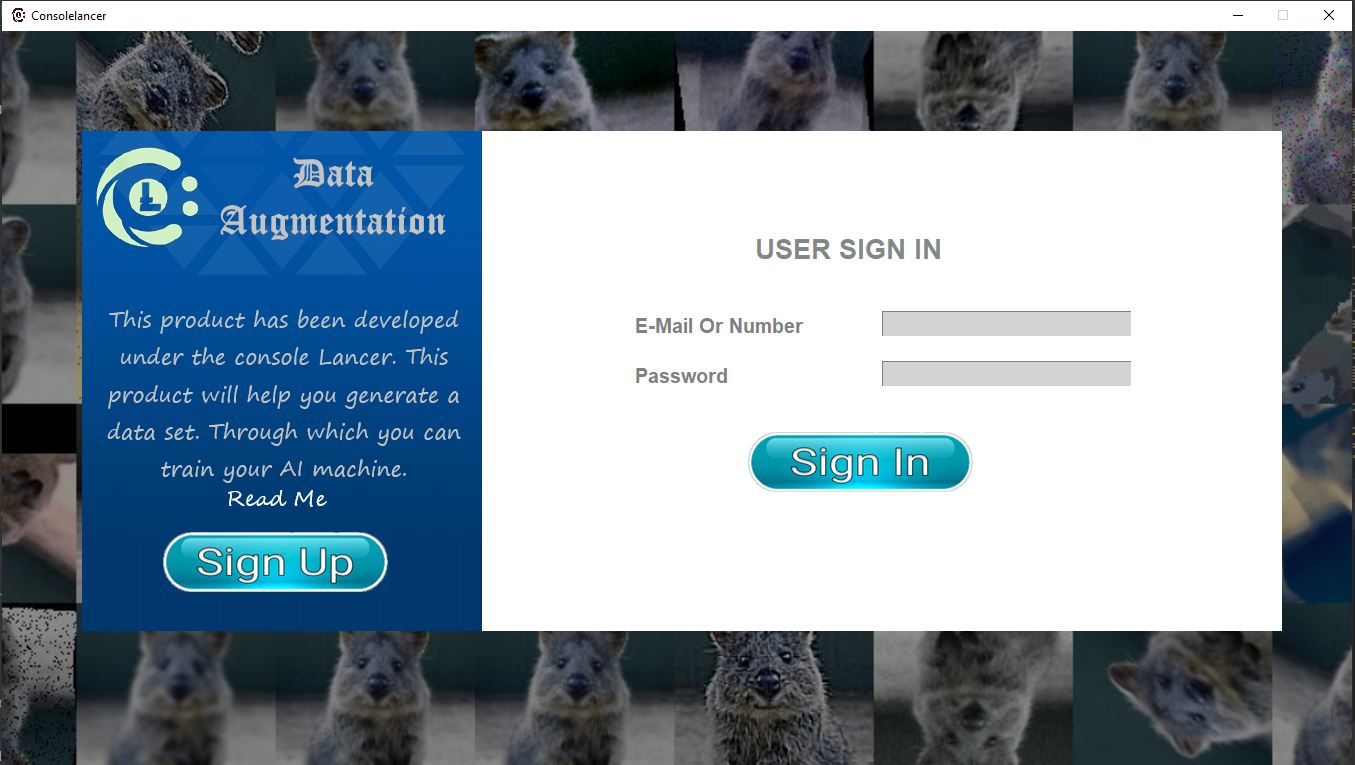
Admin Sign In

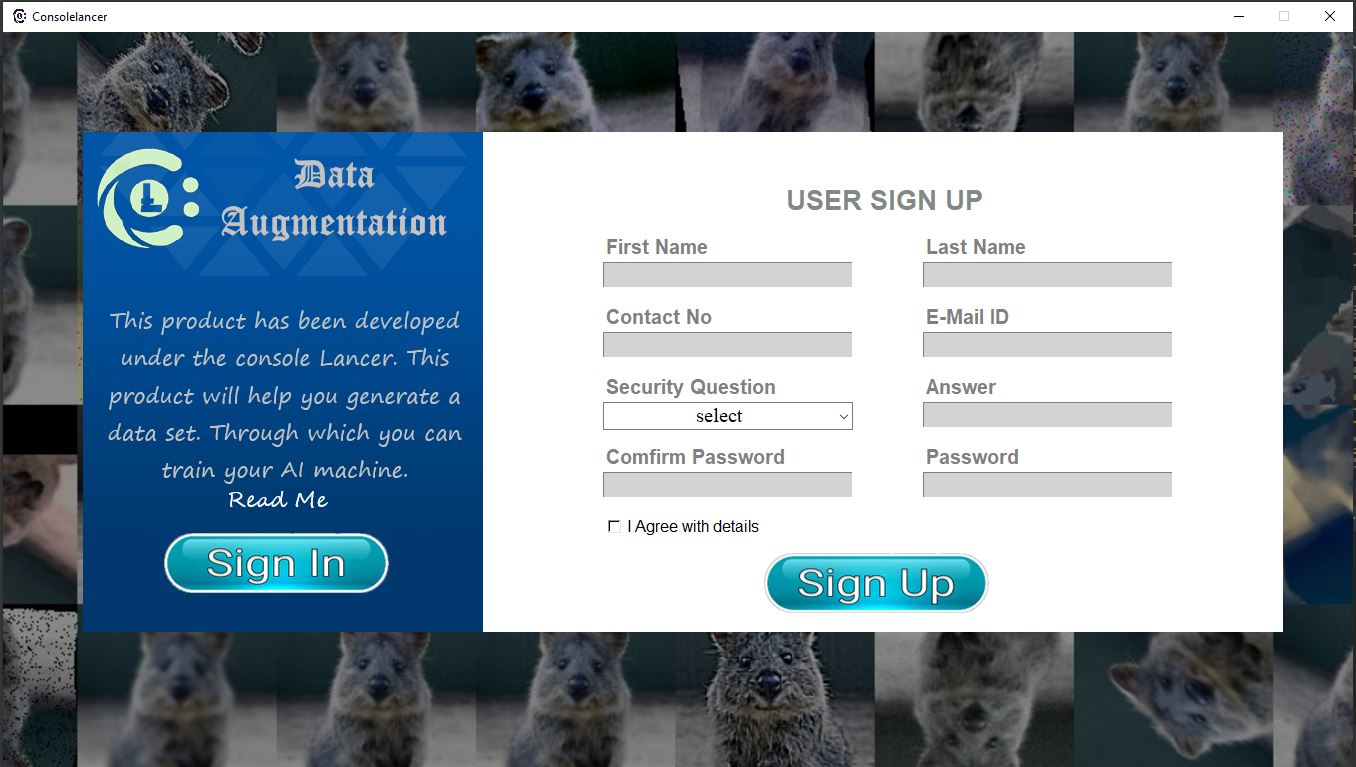
User Sign In

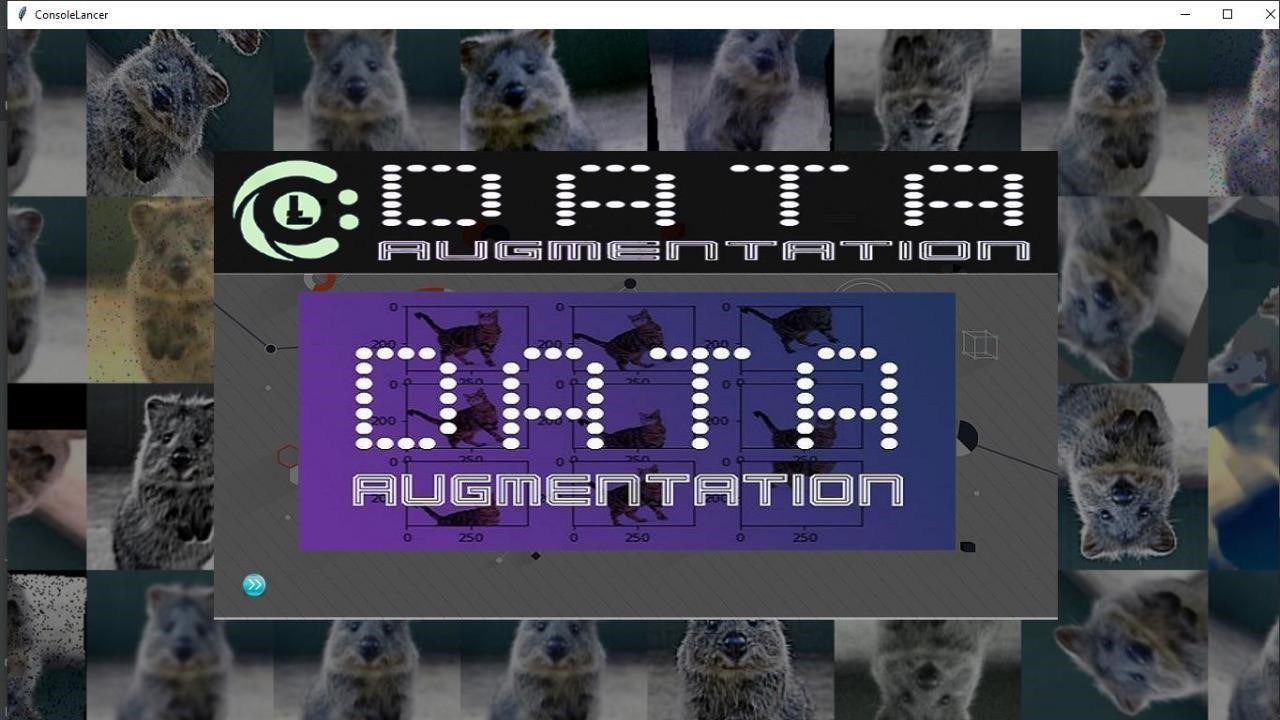
Admin Sign Up

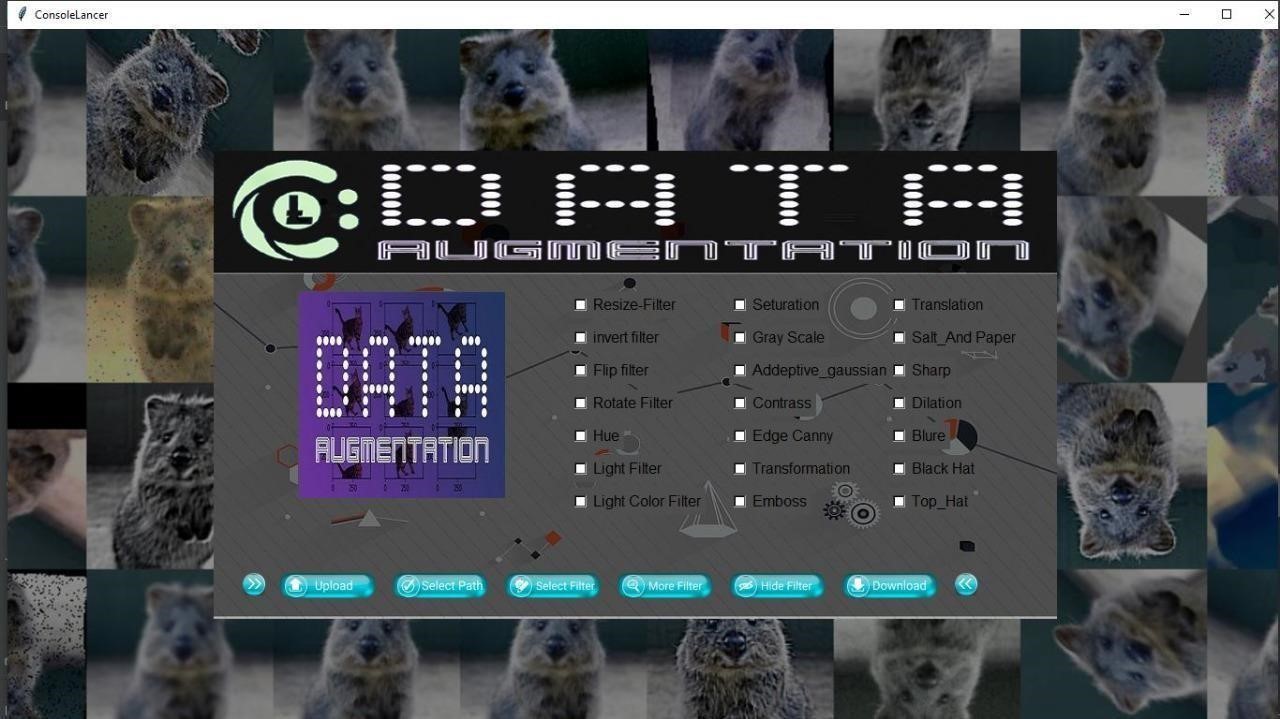


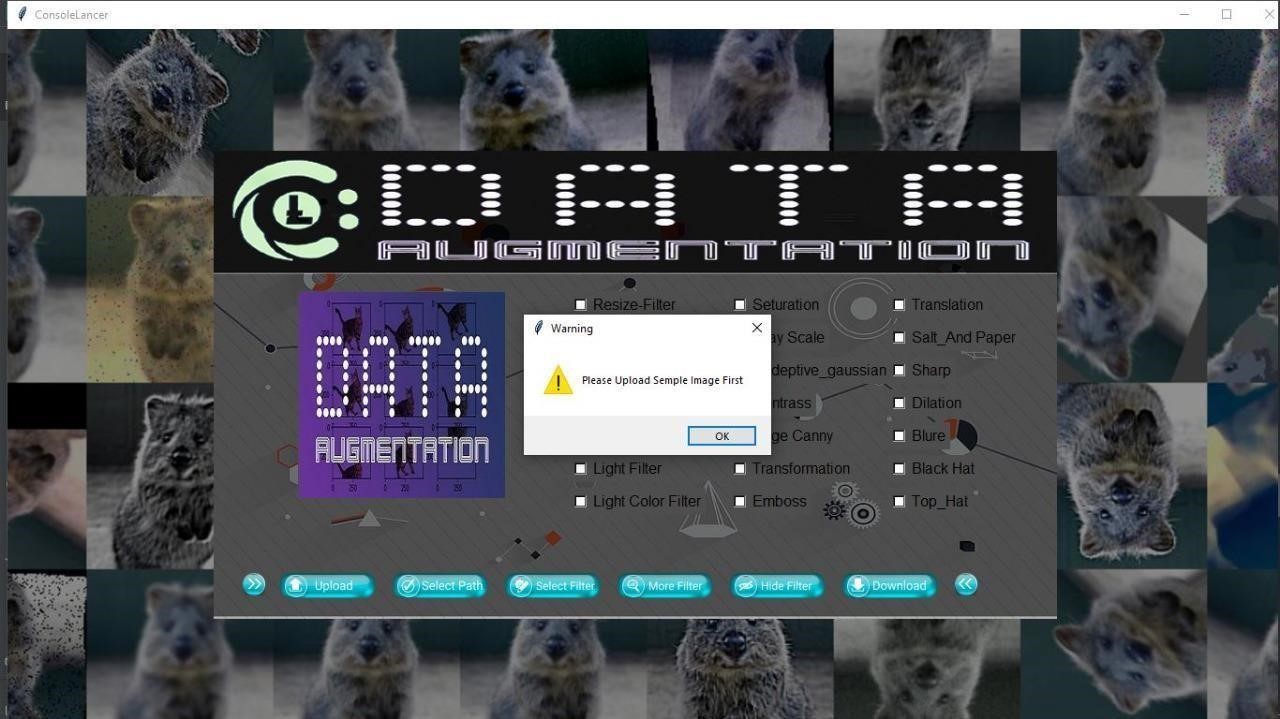
##### SCREENSHOTS

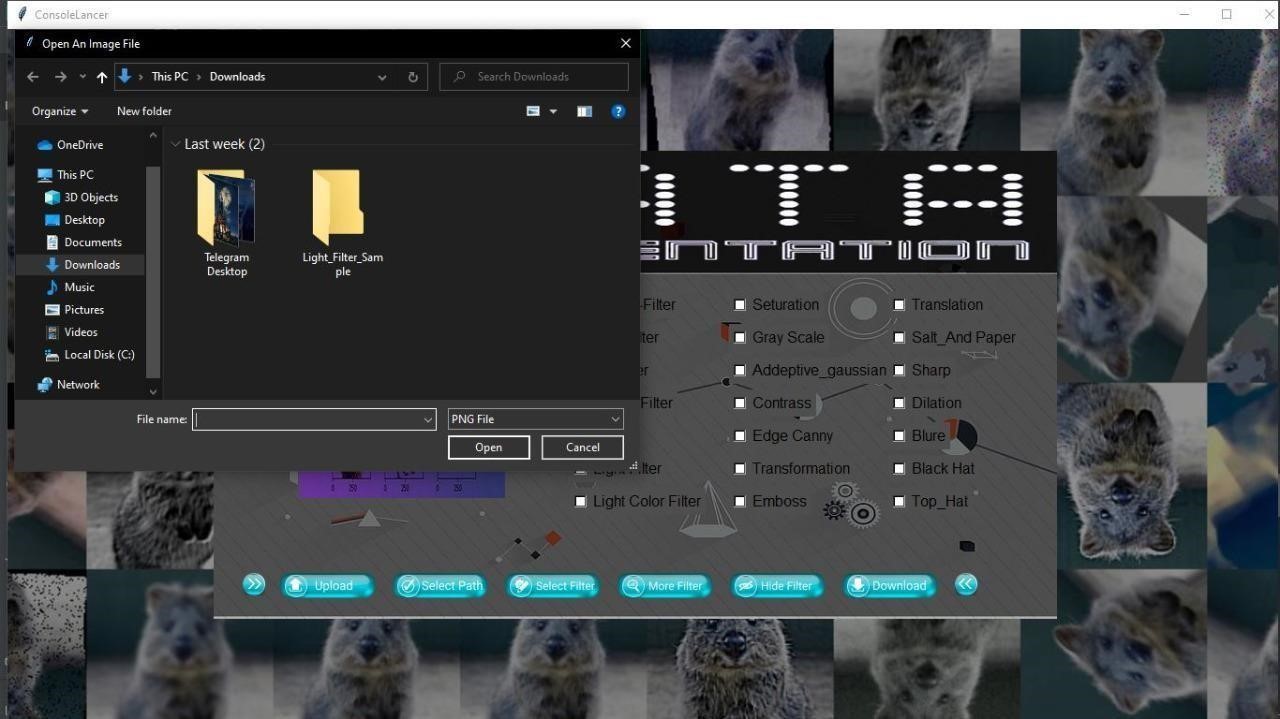


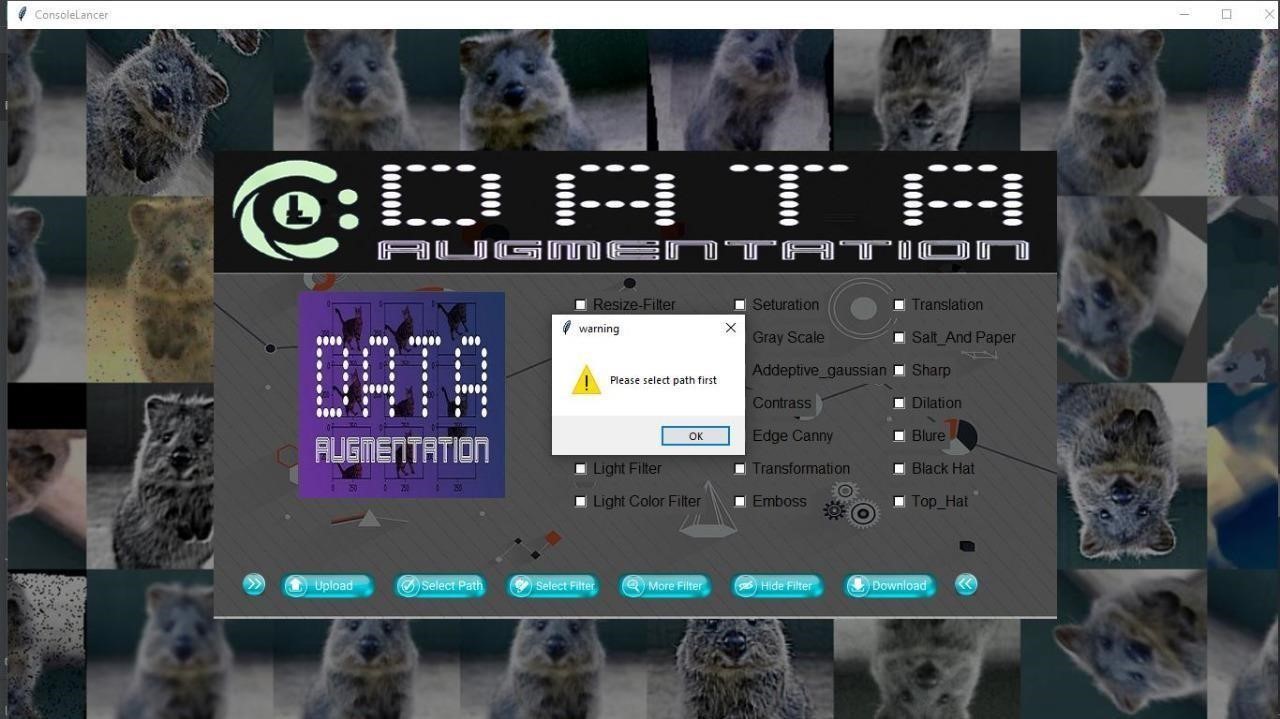


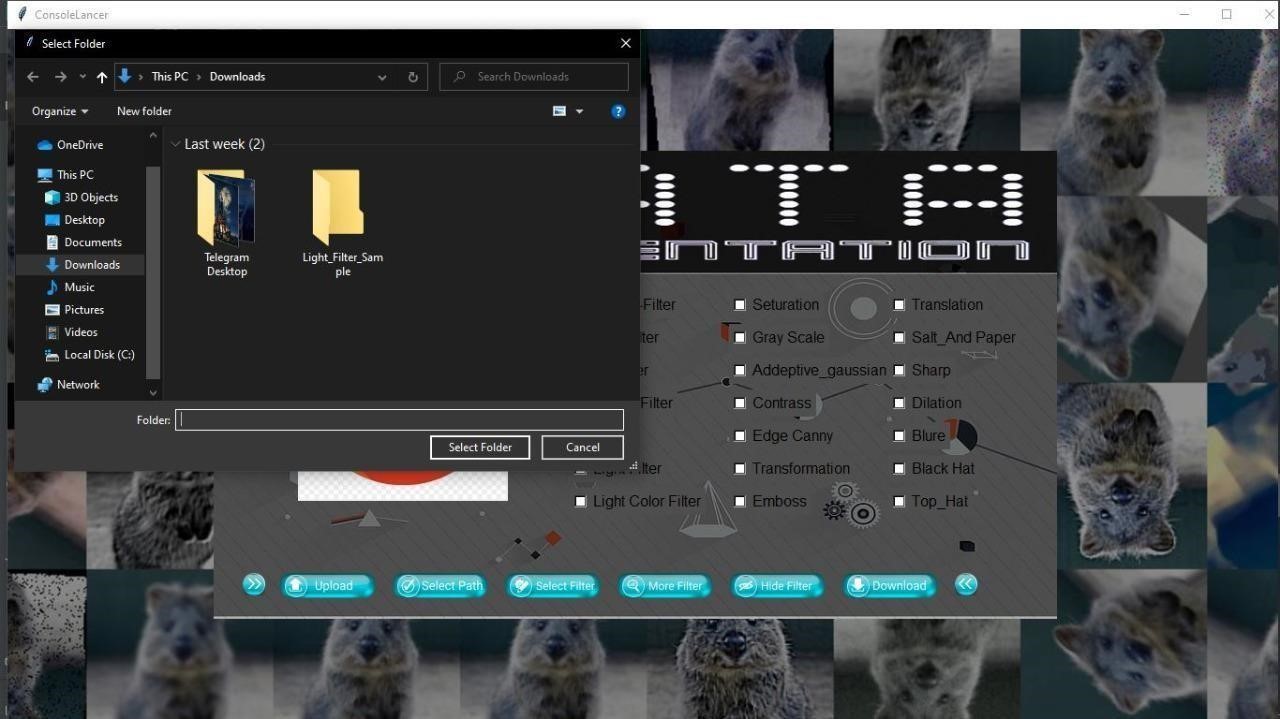


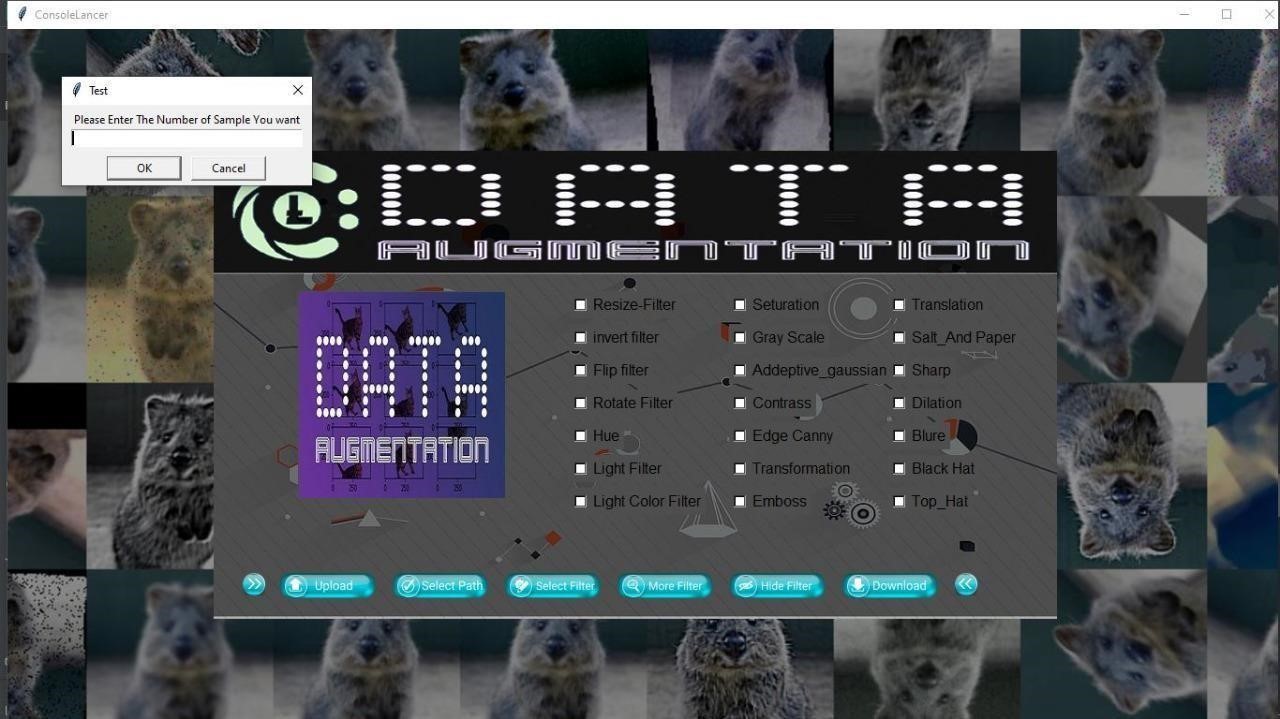


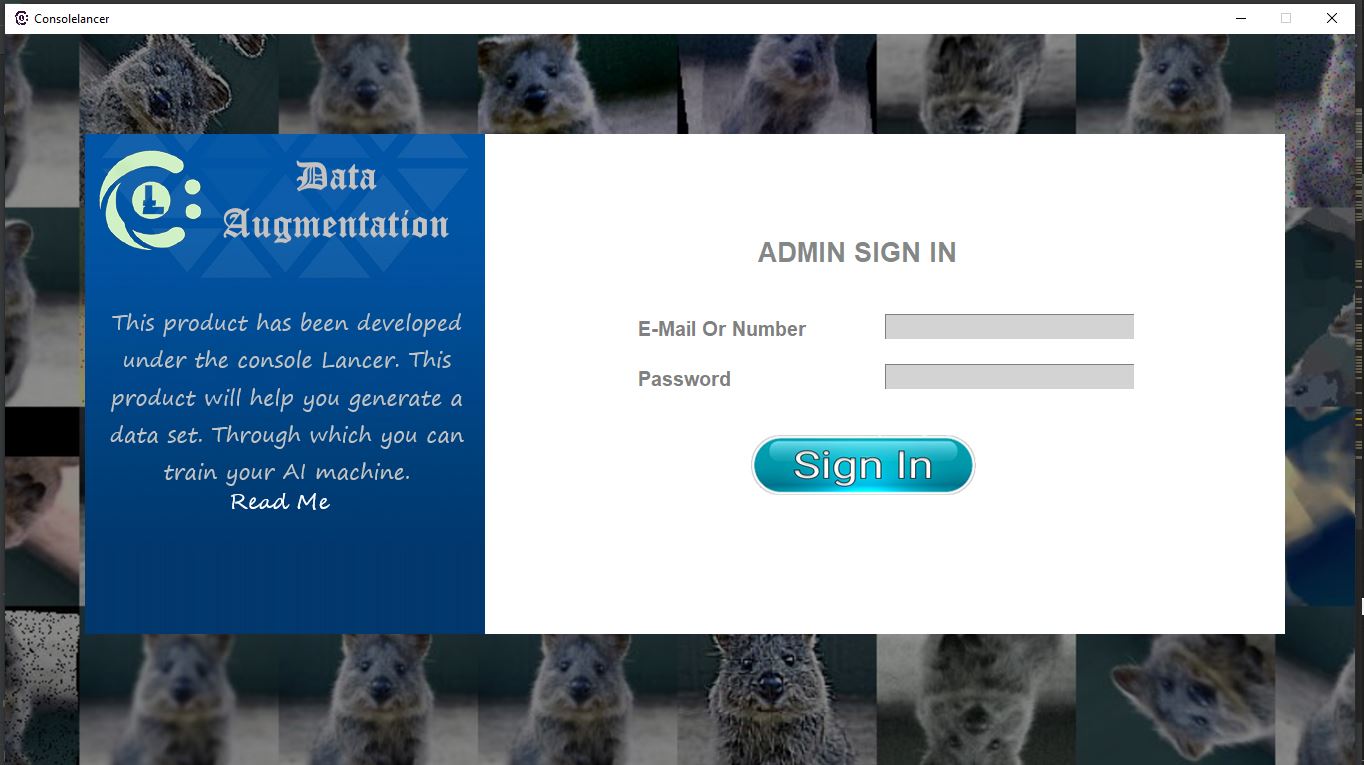




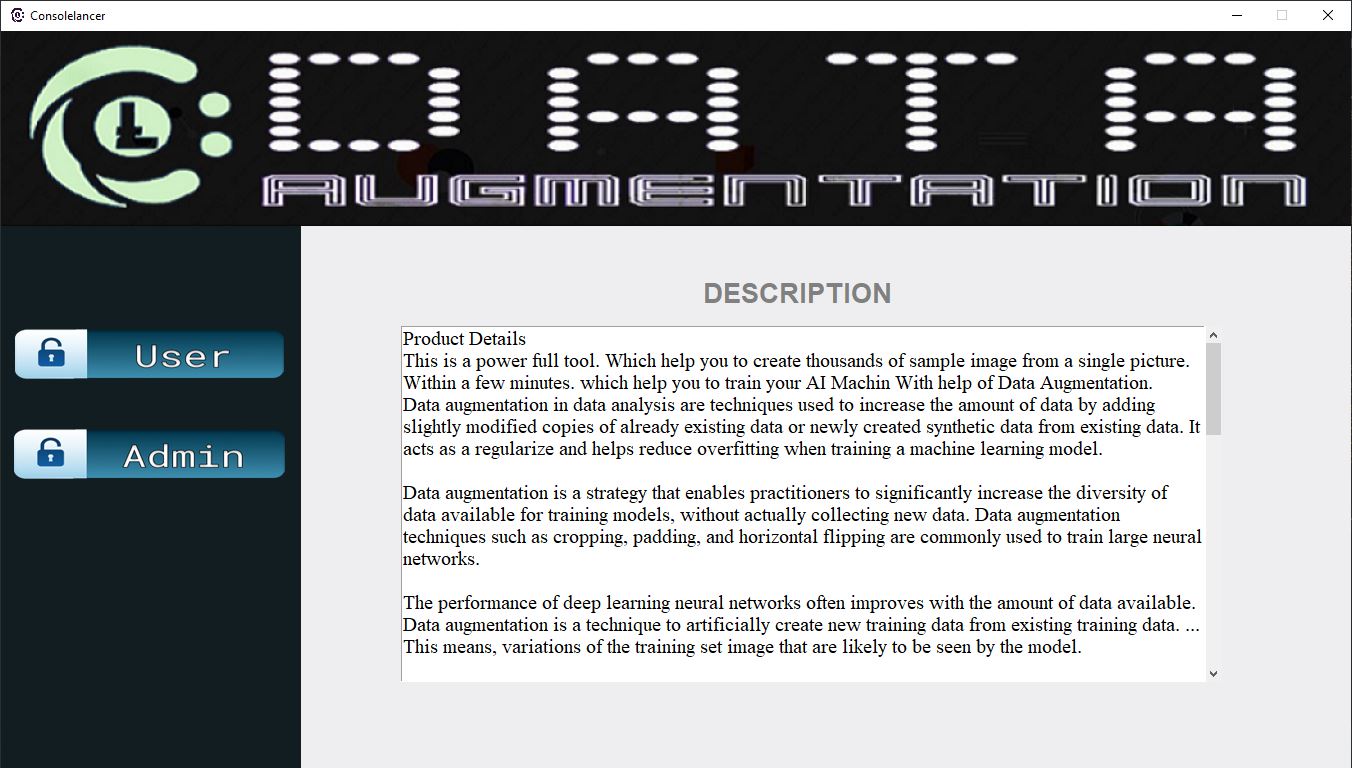












**CODING**

##### SAVE USER INFO

# Writing to an excel

# sheet using Python

import xlwt

from xlwt import Workbook

# Workbook is created

wb = Workbook()

# add\_sheet is used to create sheet.

sheet1 = wb.add\_sheet('Sheet 1')

sheet1.write(1, 0, 'ISBT DEHRADUN') # 1 is used for colums

sheet1.write(2, 0, 'SHASTRADHARA')

sheet1.write(3, 0, 'CLEMEN TOWN')

sheet1.write(4, 0, 'RAJPUR ROAD')

sheet1.write(5, 0, 'CLOCK TOWER')

sheet1.write(0, 1, 'ISBT DEHRADUN') # 1 is used for rows

sheet1.write(0, 2, 'SHASTRADHARA')

sheet1.write(0, 3, 'CLEMEN TOWN')

sheet1.write(0, 4, 'RAJPUR ROAD')

sheet1.write(0, 5, 'CLOCK TOWER')

wb.save('User\_Info.xls')

##### Admin Login

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-------------------------------------------Importing Required Package(API)-------------------------------------

--

#----------------------------------------------------------------------------------------------------------------- from tkinter import Label, Button from PIL import ImageTk

from tkinter import messagebox, Frame, Entry, END, Tk import pymysql class Register:

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-----------------------------------------------------FRONT END CODE-------------------------------------------

---

#-----------------------------------------------------------------------------------------------------------------

#=====Function=========== def \_\_init\_\_(self,root): self.root=root self.root.title("ConsoleLancer") self.root.geometry("1600x750+0+0")

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-------------------------Frame And Background------------------- #----------------------------------------------------------------

#===main-Background===

self.bg=ImageTk.PhotoImage(file="bg.png")

bg=Label(self.root,image=self.bg).place(x=0,y=0,relwidth=1,relheight=1)

#===Sub-Background=====

self.left=ImageTk.PhotoImage(file="Sub\_bg.png")

left=Label(self.root,image=self.left).place(x=80,y=100,width=400,height=500)

#===Register freame===== frame1=Frame(self.root,bg="white")

frame1.place(x=480,y=100,width=800,height=500)

#===Form Heading=======

title=Label(frame1,text="ADMIN SIGN IN", font=("time new roman",20,"bold"),bg="white",fg="#838786")

title.place(x=270,y=100)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#---------------Entry feilds and Headings ----------------------- #----------------------------------------------------------------

# =======E-mail or Number Text And Field===========

email = Label(frame1, text="E-Mail Or Number", font=("time new roman", 15, "bold"), bg="white", fg="gray") email.place(x=150, y=180)

self.email = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.email.place(x=400, y=180, width=250)

# ==========Password Text And Field=========

passw = Label(frame1, text="Password", font=("time new roman", 15, "bold"), bg="white", fg="gray")

passw.place(x=150, y=230)

self.passw = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.passw.place(x=400, y=230, width=250)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-----------------------Buttons----------------------------------- #-----------------------------------------------------------------

# =====Regester or Signin button======

# =======Singin Button======

self.signin\_btn = ImageTk.PhotoImage(file="Sign\_In.png") sigin =

Button(frame1,image=self.signin\_btn,activebackground="#ffffff",command=self.log\_in,bd=0,bg="#ff ffff",cursor="hand2")

sigin.place(x=265, y=300)

# =======Signup Button======

self.signup\_btn = ImageTk.PhotoImage(file="Sign\_Up.png") sigup =

Button(self.root,image=self.signup\_btn,activebackground="#013c74",command=self.sign\_up,bg="#0

13a71",cursor="hand2",bd=0)

#sigup.place(x=160,y=500)

#=======Read More Button======

self.read\_more = ImageTk.PhotoImage(file="Read\_more.png")

read\_more = Button(self.root, text="Read More",activebackground="#013c74",

image=self.read\_more, command=self.read\_More\_page, bg="#013c74", bd=0, cursor="hand2")

read\_more.place(x=220, y=455)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-----------------------------------------------BACK END CODE--------------------------------------------------

-----

#------------------------------------------------------------------------------------------------------------------- #========Function for going to sign in page============= def log\_in(self): if self.email.get()=="" or self.passw.get()=="":

messagebox.showerror("Error","Please Enter User Name And Password",parent=self.root) else: try:

con=pymysql.connect(host="localhost",user="root",password="",database="aug") cur=con.cursor()

cur.execute("select \* from adminTab where admin=%s and password=%s",(self.email.get(),self.passw.get()))

row=cur.fetchone() if row==None:

messagebox.showerror("Error","Invalid User Name and Password",parent=self.root) else:

messagebox.showinfo("Success","Welcome",parent=self.root) self.root.destroy() import Admin\_panel con.close() except Exception as es: messagebox.showerror("Error",f"Error Due to: {str(es)}",parent=self.root)

# =======Function for Going to Sign up page======================== def sign\_up(self): self.root.destroy()

import Admin\_Registration

#====Function to delete current page and jump on Read More Page===== def read\_More\_page(self):

self.root.destroy()

import Read\_More

# ==========function to clear the fields after success=============== def clear(self):

self.email.delete(0, END)

self.passw.delete(0, END)

root=Tk() obj=Register(root) root.mainloop()

##### Admin DashBoard

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-------------------------------------------Importing Required Package(API)-------------------------------------

--

#-----------------------------------------------------------------------------------------------------------------

import mysql from PIL import ImageTk import tkinter as tk import mysql.connector from tkinter import ttk, filedialog

from xlwt import Workbook

class Register:

#

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -----------------------------------------------------FRONT END CODE------------------------------------------

----

# -----------------------------------------------------------------------------------------------------------------

#=====Root Function=========== def \_\_init\_\_(self,root):

self.root=root self.root.title("ConsoleLancer")

self.root.geometry("1350x740+0+0")

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -------------------------Frame And Background-------------------

# ----------------------------------------------------------------

# ===========================Frames===============================

# -------------------First Frame----------------------------------- self.topLeft = ImageTk.PhotoImage(file="Admin\_TopLeft.png")

topleft = tk.Label(self.root, image=self.topLeft) topleft.place(x=-2, y=0)

# -------------------Second Frame---------------------------------- self.left = ImageTk.PhotoImage(file="Admin\_panel\_Header.png") left = tk.Label(self.root, image=self.left)

left.place(x=300, y=0, width=1050, height=195)

# -------------------Third Frame---------------------------------- frame3 = tk.Frame(self.root,bd=2,bg="#141F23") frame3.place(x=-5, y=250, width=310, height=547)

# -------------------Fourth Frame---------------------------------- self.frame4 = tk.Frame(self.root,bg="#3b3f42")

self.frame4.place(x=300, y=195, width=1050, height=547)

#=====================Blank Area [frame 1]=========================

#-------Connecting To DataBase For Printing Admin Name------------- mydbadmin = mysql.connector.connect(user="root", password="", database="aug", host="localhost")

cursoradmin = mydbadmin.cursor() sql = "SELECT `admin` FROM `adminTab`"

#-----------------------Fetching Admin Name------------------------- cursoradmin.execute(sql) adminName = cursoradmin.fetchone() for i in adminName: x = adminName

#-----------------Printing Admin Name On Deshboard------------------

self.admin\_Name = tk.Label(self.root, text="%s"%adminName, font=("time new roman",15,

"bold"), bg="#141F23",

fg="#838786")

self.admin\_Name.place(x=75,y=190)

#-----------------------Sepration Line------------------------------ self.line = ImageTk.PhotoImage(file="Line.png")

line = tk.Label(self.root, image=self.line,bg="#293f4c", width=260) line.place(x=20, y=230)

'''This frame is holding Admin Menu bar Title'''

#=====================Admin Panel Heading [Frame 2]================

'''This frame is containing Header. Which is already decleared & initialized in above code'''

#=====================Button Area [Frame 3]========================

#-------------Creating and placing Show Feature Button--------------- self.show = ImageTk.PhotoImage(file="Feature.png") self.show2 =

tk.Button(frame3,image=self.show,activebackground="#172637",width=400,bg="#1a262b", command=self.show\_Feature, bd=0, cursor="hand2")

self.show2.place(x=-100, y=25)

#------------------Creating User Information button-------------------

self.user\_Info\_btt = tk.Button(frame3, text="User

Info",activebackground="#172637",font=("time new roman",15,"bold"),fg="white",bg="#1a262b", command=self.user\_Info, bd=0, cursor="hand2")

#-----Creating Download button for Downloading user information------- self.show5 = ImageTk.PhotoImage(file="Feature.png") self.download\_btt =

tk.Button(frame3,text="Download",activebackground="#172637",font=("time new roman",15,"bold"),fg="white",bg="#293f4c", command=self.download, bd=0, cursor="hand2") self.download1\_btt = tk.Button(frame3, text="- Download Info",activebackground="#172637", font=("time new roman", 15, "bold"), fg="white",

bg="#293f4c", command=self.download1, bd=0, cursor="hand2")

#--------------------Creating Sign out button------------------------- self.admin\_Sign\_out\_btt = tk.Button(frame3, text="Sign Out",

activebackground="#172637",font=("time new

roman",15,"bold"),fg="white",bg="#293f4c",command=self.sign\_out,bd=0, cursor="hand2")

#----------------------Creating hide button--------------------------- self.show5 = ImageTk.PhotoImage(file="Hide\_Feature.png")

self.hide = tk.Button(frame3,image=self.show5,activebackground="#172637", font=("time new roman", 15, "bold"), fg="white",width=400,bg="#1a262b",

command=self.hide\_Feature, bd=0, cursor="hand2")

#========================Download Info=================================

#------------Creating button for downloading all details--------------- self.download\_All\_btt = tk.Button(frame3,

text="All",activebackground="#172637",command=self.user\_All, font=("time new roman", 15,

"bold"),

fg="white", bg="#293f4c", bd=0, cursor="hand2")

#-------------Creating button for downloading name of users-------------

self.download\_Name\_btt = tk.Button(frame3, text="Name",activebackground="#172637", font=("time new roman", 15, "bold"),

fg="white", bg="#293f4c", command=self.user\_Name, bd=0, cursor="hand2")

#------------Creating button for downloading number -------------------

self.download\_Number\_btt = tk.Button(frame3, text="Number",activebackground="#172637", font=("time new roman", 15, "bold"),

fg="white", bg="#293f4c", command=self.user\_Number, bd=0, cursor="hand2")

#------------Creating button for downloading numbers-------------------

self.download\_Id\_btt = tk.Button(frame3, text="User ID",activebackground="#172637", font=("time new roman", 15, "bold"), fg="white",

bg="#293f4c", command=self.user\_ID, bd=0, cursor="hand2")

# ====================Description Area [Frame 4]====================

#---------------Placing Headig in frame 4 --------------------------

self.title = tk.Label(self.frame4, text="USER INFORMATION",font=("time new roman", 20, "bold"), bg="#3b3f42" , fg="white")

#----------------Placing default image in frame 4 -------------------- self.welcomeimg = ImageTk.PhotoImage(file="Admin\_Welcome.png") self.welcome = tk.Label(self.frame4, image=self.welcomeimg,bg="#3b3f42" ) self.welcome.place(x=300, y=50)

'''Form 4 code is working for Back End'''

#

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -----------------------------------------------BACK END CODE-------------------------------------------------

------

# -------------------------------------------------------------------------------------------------------------------

#========================Creating function for showing feature================== def show\_Feature(self):

self.show2.place\_forget()

self.hide.place(x=-100, y=25)

self.admin\_Sign\_out\_btt.place(x=85, y=60) self.user\_Info\_btt.place(x=85, y=95) self.download\_btt.place(x=70, y=130)

def download(self): self.download\_btt.place\_forget()

self.download1\_btt.place(x=70, y=130)

self.download\_All\_btt.place(x=100,y=165) self.download\_Name\_btt.place(x=100,y=200) self.download\_Number\_btt.place(x=100, y=235) self.download\_Id\_btt.place(x=100,y=270)

def download1(self): self.download1\_btt.place\_forget()

self.download\_btt.place(x=70, y=130)

self.download\_All\_btt.place\_forget() self.download\_Name\_btt.place\_forget() self.download\_Number\_btt.place\_forget() self.download\_Id\_btt.place\_forget()

def hide\_Feature(self): self.hide.place\_forget() self.show2.place(x=-100, y=25) self.user\_Info\_btt.place\_forget() self.download\_btt.place\_forget() self.admin\_Sign\_out\_btt.place\_forget() self.download1\_btt.place\_forget() self.download\_All\_btt.place\_forget() self.download\_Name\_btt.place\_forget() self.download\_Number\_btt.place\_forget() self.download\_Id\_btt.place\_forget()

#=========================Function for User Registration================ def user\_Info(self): self.welcome.place\_forget() self.title.place(x=400, y=80)

mydb = mysql.connector.connect(user="root", password ="", database="aug", host="localhost") cursor = mydb.cursor()

sql = "SELECT `first\_name`, `last\_name`, `phone\_no`, `mail` FROM `user`"

cursor.execute(sql) rows = cursor.fetchall()

total = cursor.rowcount

tv = ttk.Treeview(self.frame4, columns = (1,2,3,4), show = "headings", height ="8") tv.place(x=120,y=150)

tv.heading(1, text="First Name") tv.heading(2, text="Last Name") tv.heading(3, text="Phone Number") tv.heading(4, text="Email Id")

for i in rows:

tv.insert('', 'end', values = i)

cursor.close()

mydb.close

def user\_All(self): nameFilePath = filedialog.askdirectory(parent=root,initialdir="/path/to/start/",title='Please select a directory')

if nameFilePath=="":

pass else:

#=============================Fetching First Name================================

mydb1 = mysql.connector.connect(user="root", password="", database="aug", host="localhost")

cursor1 = mydb1.cursor()

sql = "SELECT `first\_name` FROM `user`"

cursor1.execute(sql) rows1 = cursor1.fetchall() #total = cursor1.rowcount wb = Workbook()

sheet1 = wb.add\_sheet('Sheet 1') sheet1.write(0, 0,'First Name') sheet1.write(0, 1, 'Last Name') sheet1.write(0, 2, 'Phone Number') sheet1.write(0, 3, 'Gmail ID')

row\_no1 = 1 for i in rows1:

sheet1.write(row\_no1, 0, "%s" % i) # 1 is used for rows row\_no1 = row\_no1 + 1

cursor1.close()

mydb1.close()

#=========================Fetching Last Name

===============================

mydb2 = mysql.connector.connect(user="root", password="", database="aug", host="localhost")

cursor2 = mydb2.cursor()

sql = "SELECT `last\_name` FROM `user`"

cursor2.execute(sql) rows2 = cursor2.fetchall() row\_no2 = 1 for j in rows2:

sheet1.write(row\_no2, 1, "%s" % j) # 1 is used for rows row\_no2 = row\_no2 + 1

cursor2.close()

mydb2.close()

#==========================Fetching

Number========================================

mydb3 = mysql.connector.connect(user="root", password="", database="aug", host="localhost")

cursor3 = mydb3.cursor()

sql = "SELECT `phone\_no` FROM `user`"

cursor3.execute(sql)

rows3 = cursor3.fetchall()

row\_no3 = 1 for j in rows3:

sheet1.write(row\_no3, 2, "%s" % j) # 1 is used for rows row\_no3 = row\_no3 + 1

cursor3.close()

mydb3.close()

#=============================Fetching Gmail

ID====================================

mydb4 = mysql.connector.connect(user="root", password="", database="aug", host="localhost")

cursor4 = mydb4.cursor()

sql = "SELECT `mail` FROM `user`"

cursor4.execute(sql)

rows4 = cursor4.fetchall()

row\_no4 = 1 for j in rows4:

sheet1.write(row\_no4, 3, "%s" % j) # 1 is used for rows row\_no4 = row\_no4 + 1

cursor4.close() mydb4.close()

wb.save('%s/All Information.xls' % nameFilePath)

def user\_Name(self): nameFilePath = filedialog.askdirectory(parent=root,initialdir="/path/to/start/",title='Please select a directory')

if nameFilePath=="":

pass else:

mydb1 = mysql.connector.connect(user="root", password="", database="aug", host="localhost")

cursor1 = mydb1.cursor()

sql = "SELECT `first\_name` FROM `user`"

cursor1.execute(sql) rows = cursor1.fetchall() total = cursor1.rowcount

wb = Workbook()

sheet1 = wb.add\_sheet('Sheet 1') row\_no =1 for i in rows:

sheet1.write(row\_no,0,"%s"% i) # 1 is used for rows row\_no = row\_no + 1

cursor1.close()

mydb1.close()

mydb2 = mysql.connector.connect(user="root", password="", database="aug", host="localhost")

cursor2 = mydb2.cursor()

sql = "SELECT `last\_name` FROM `user`"

cursor2.execute(sql)

rows2 = cursor2.fetchall()

row\_no1 = 1 for j in rows2:

sheet1.write(row\_no1,1, "%s" % j) # 1 is used for rows row\_no1 = row\_no1 + 1

cursor2.close() mydb2.close()

wb.save('%s/User\_Name.xls' % nameFilePath)

def user\_Number(self):

numberFilePath = filedialog.askdirectory(parent=root,initialdir="/path/to/start/",title='Please select a directory')

if numberFilePath=="":

pass

else:

mydb3 = mysql.connector.connect(user="root", password="", database="aug", host="localhost")

cursor3 = mydb3.cursor()

sql = "SELECT `phone\_no` FROM `user`"

cursor3.execute(sql) rows3 = cursor3.fetchall() total = cursor3.rowcount

wb = Workbook()

sheet3 = wb.add\_sheet('Sheet 1') row\_no = 1 for i in rows3:

sheet3.write(row\_no, 0, "%s" % i) # 1 is used for rows row\_no = row\_no + 1

cursor3.close()

mydb3.close()

wb.save('%s/User\_Number.xls' % numberFilePath)

def user\_ID(self):

idFilePath = filedialog.askdirectory(parent=root,initialdir="/path/to/start/",title='Please select a directory')

if idFilePath == "":

pass else:

mydb4 = mysql.connector.connect(user="root", password="", database="aug", host="localhost")

cursor4 = mydb4.cursor()

sql = "SELECT `mail` FROM `user`"

cursor4.execute(sql) rows4 = cursor4.fetchall() total = cursor4.rowcount wb = Workbook()

sheet1 = wb.add\_sheet('Sheet 1') row\_no = 1 for i in rows4:

sheet1.write(row\_no, 0, "%s" % i) # 1 is used for rows row\_no = row\_no + 1 cursor4.close()

mydb4.close()

wb.save('%s/User\_ID.xls' % idFilePath)

#==================This function is working for achiving Logout Functionality================== def sign\_out(self): self.root.destroy()

import Admin\_login

root= tk.Tk() obj=Register(root) root.resizable(False, False)

root.mainloop()

##### User Registration

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-------------------------------------------Importing Required Package(API)-------------------------------------

--

#----------------------------------------------------------------------------------------------------------------- from tkinter import Label

from PIL import ImageTk

from tkinter import ttk, messagebox, Frame, Entry, CENTER, IntVar, Checkbutton, Button, END, Tk import pymysql class Register: #

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -----------------------------------------------------FRONT END CODE------------------------------------------

----

# -----------------------------------------------------------------------------------------------------------------

def \_\_init\_\_(self,root): self.root=root self.root.title("ConsoleLancer") self.root.geometry("1600x750+0+0")

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -------------------------Frame And Background-------------------

# ----------------------------------------------------------------

#main-Background

self.bg=ImageTk.PhotoImage(file="bg.png")

bg=Label(self.root,image=self.bg).place(x=0,y=0,relwidth=1,relheight=1)

#Sub-Background

self.left=ImageTk.PhotoImage(file="Sub\_bg.png")

left=Label(self.root,image=self.left).place(x=80,y=100,width=400,height=500)

#===Register freame=== frame1=Frame(self.root,bg="white")

frame1.place(x=480,y=100,width=800,height=500)

#=====Form Area =====

title=Label(frame1,text="USER SIGN UP", font=("time new roman",20,"bold") ,bg="white"

,fg="#838786") title.place(x=300,y=50)

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# ---------------Entry feilds and Headings -----------------------

# ----------------------------------------------------------------

# =======First Name Text And Field===========

f\_name = Label(frame1, text="First Name", font=("time new roman", 15, "bold"), bg="white", fg="gray")

f\_name.place(x=120, y=100)

self.fname = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.fname.place(x=120, y=130, width=250)

# ==========Last Name Text And Field=========

l\_name = Label(frame1, text="Last Name", font=("time new roman", 15, "bold"), bg="white", fg="gray")

l\_name.place(x=440, y=100)

self.lname = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.lname.place(x=440, y=130, width=250)

# =======Contact No.===========

contact = Label(frame1, text="Contact No", font=("time new roman", 15, "bold"), bg="white", fg="gray").place( x=120, y=170)

self.contact = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.contact.place(x=120, y=200, width=250)

# ==========E-Mail Id=========

E\_mail = Label(frame1, text="E-Mail ID", font=("time new roman", 15, "bold"), bg="white", fg="gray").place( x=440, y=170)

self.e\_mail = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.e\_mail.place(x=440, y=200, width=250)

# =======Security Quistion.===========

ques = Label(frame1, text="Security Question", font=("time new roman", 15, "bold"), bg="white",

fg="gray").place(x=120, y=240)

# ======Combo Box===============

self.select = ttk.Combobox(frame1, font=("times new roman", 15), state='readonly', justify=CENTER)

self.select['values'] = ("select", "Your first place ", "Your Best friend Name ",) self.select.place(x=120, y=270, width=250) self.select.current(0)

# ==========Asnwer ========= ans = Label(frame1, text="Answer", font=("time new roman", 15, "bold"), bg="white", fg="gray").place( x=440, y=240)

self.ans = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.ans.place(x=440, y=270, width=250)

# =====Password=====

psw = Label(frame1, text="Password", font=("time new roman", 15, "bold"), bg="white", fg="gray").place(x=440, y=310)

self.psw = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.psw.place(x=120, y=340, width=250)

# =======Confirm Password===========

cpsw = Label(frame1, text="Comfirm Password", font=("time new roman", 15, "bold"), bg="white", fg="gray") cpsw.place(x=120, y=310)

self.cpsw = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.cpsw.place(x=440, y=340, width=250)

# =====Check Box===========

self.check=IntVar()

chk = Checkbutton(frame1, text="I Agree with details", variable=self.check, onvalue=1, offvalue=0, bg="white",font=("time new roman", 12)) chk.place(x=120, y=380)

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -----------------------Buttons----------------------------------- # -----------------------------------------------------------------

# =====Regester button======

#=======Singin Button======

self.signin\_btn = ImageTk.PhotoImage(file="Sign\_In.png") sigin =

Button(self.root,image=self.signin\_btn,activebackground="#013a71",command=self.sign\_in,bg="#01 3a71",bd=0,cursor="hand2").place(x=160,y=500)

#=======Singup Button======

self.signup\_btn = ImageTk.PhotoImage(file="Sign\_Up.png") sigup = Button(frame1, image=self.signup\_btn,cursor="hand2",activebackground="#ffffff",bd=0,bg="#ffffff",command=self.

register\_data)

sigup.place(x=280, y=420)

# =======Read More Button======

self.read\_more = ImageTk.PhotoImage(file="Read\_more.png") read\_more = Button(self.root, text="Read More",activebackground="#013c74",image=self.read\_more,

command=self.read\_More\_page,bg="#013c74", bd=0, cursor="hand2").place(x=220, y=455)

#

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -----------------------------------------------BACK END CODE-------------------------------------------------

------

# -------------------------------------------------------------------------------------------------------------------

#==========Function for regestration form data insertion and fetch========= def register\_data(self): if self.fname.get()=="" or self.lname.get()=="" or self.contact.get()==""or self.e\_mail.get()==""or self.select.get()==""or self.ans.get()==""or self.psw.get()==""or self.cpsw.get()=="": messagebox.showerror("Error","All fields are required ",parent=self.root) elif self.psw.get()!=self.cpsw.get():

messagebox.showerror("Error", "Password must be same ", parent=self.root) elif self.check.get()==0: messagebox.showerror("Error", "agree check our tems and condition ", parent=self.root) else: try:

con=pymysql.connect(host="localhost",user="root",password="",database="aug") cur=con.cursor()

cur.execute("select \* from user where phone\_no=%s", self.contact.get()) prow = cur.fetchone()

cur.execute("select \* from user where mail=%s", self.e\_mail.get()) row=cur.fetchone() if row!=None or prow!=None: messagebox.showerror("Error", "Email or phone no already registered try with another one ", parent=self.root) else:

cur.execute("insert into

user(first\_name,last\_name,phone\_no,mail,Ques,answer,password) values(%s,%s,%s,%s,%s,%s,%s)",

(

self.fname.get(), self.lname.get(), self.contact.get(), self.e\_mail.get(), self.select.get(), self.ans.get(), self.psw.get(),

))

con.commit() #Data Inserted con.close() #connection closed

messagebox.showinfo("success", "Register Success",parent=self.root)

self.clear() self.root.destroy() import Aug except Exception as es: messagebox.showerror("Error",f"Error Due to {str(es)}", parent=self.root)

# =======Function for going to sign in page======

# =======Function for going to sign in page====== def sign\_in(self): self.root.destroy()

import User\_Login

# ==========function to clear the fields after success=============== def clear(self):

self.fname.delete(0, END) self.lname.delete(0, END) self.contact.delete(0, END) self.e\_mail.delete(0, END) self.ans.delete(0,END) self.cpsw.delete(0,END) self.select.current(0)

self.psw.delete(0, END)

# ====Function to delete current page and jump on Read More Page======== def read\_More\_page(self):

self.root.destroy()

import Read\_More

root=Tk() obj=Register(root) root.mainloop()

##### User Login

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-------------------------------------------Importing Required Package(API)-------------------------------------

--

#-----------------------------------------------------------------------------------------------------------------

from tkinter import \* from PIL import ImageTk from tkinter import messagebox import pymysql class Register: #

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -----------------------------------------------------FRONT END CODE------------------------------------------

----

# -----------------------------------------------------------------------------------------------------------------

#=====Root Function=========== def \_\_init\_\_(self,root): self.root=root self.root.title("ConsoleLancer") self.root.geometry("1600x750+0+0")

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -------------------------Frame And Background------------------- # ----------------------------------------------------------------

#===========main-Background============= self.bg=ImageTk.PhotoImage(file="bg.png")

bg=Label(self.root,image=self.bg).place(x=0,y=0,relwidth=1,relheight=1)

#===========Sub-Background============== self.left=ImageTk.PhotoImage(file="Sub\_bg.png")

left=Label(self.root,image=self.left).place(x=80,y=100,width=400,height=500)

#===========Register freame============== frame1=Frame(self.root,bg="white")

frame1.place(x=480,y=100,width=800,height=500)

#============Form Area ================== title=Label(frame1,text="USER SIGN IN", font=("time new roman",20,"bold"),bg="white",fg="#838786") title.place(x=270,y=100)

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# ---------------Entry feilds and Headings -----------------------

# ----------------------------------------------------------------

#=======E-mail or Number Text And Field===

email = Label(frame1, text="E-Mail Or Number", font=("time new roman", 15, "bold"), bg="white", fg="gray") email.place(x=150, y=180)

self.email = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.email.place(x=400, y=180, width=250)

#==========Password Text And Field=========

passw = Label(frame1, text="Password", font=("time new roman", 15, "bold"), bg="white", fg="gray")

passw.place(x=150, y=230)

self.passw = Entry(frame1, font=("times new roman", 15), bg="lightgray") self.passw.place(x=400, y=230, width=250)

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -----------------------Buttons----------------------------------- # -----------------------------------------------------------------

#=======Signup Button======

self.signin\_btn = ImageTk.PhotoImage(file="Sign\_In.png") sigin =

Button(frame1,image=self.signin\_btn,activebackground="white",command=self.log\_in,bg="#ffffff",b d=0,cursor="hand2")

sigin.place(x=265, y=300)

#=======Singin Button======

self.signup\_btn = ImageTk.PhotoImage(file="Sign\_Up.png") sigup =

Button(self.root,image=self.signup\_btn,activebackground="#013c74",command=self.sign\_up,bg="#0

13a71",cursor="hand2",bd=0)

sigup.place(x=160, y=500)

# =======Read More Button======

self.read\_more = ImageTk.PhotoImage(file="Read\_more.png")

read\_more = Button(self.root,image=self.read\_more,activebackground="#013c74", command=self.read\_More\_page, bg="#013c74",

bd=0, cursor="hand2").place(x=220, y=455)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-----------------------------------------------BACK END CODE------------------------------------------------------

#-------------------------------------------------------------------------------------------------------------------

#=======Function for Going to Sign up page============ def sign\_up(self): self.root.destroy()

import User\_Registration

#========Function for going to sign in page============= def log\_in(self): if self.email.get()=="" or self.passw.get()=="":

messagebox.showerror("Error","Please Enter User Name And Password",parent=self.root) else: try:

con=pymysql.connect(host="localhost",user="root",password="",database="aug") cur=con.cursor()

cur.execute("select \* from user where mail=%s and password=%s",(self.email.get(),self.passw.get()))

row=cur.fetchone() if row==None:

messagebox.showerror("Error","Invalid User Name and Password",parent=self.root) else:

messagebox.showinfo("Success","Welcome",parent=self.root) self.root.destroy() import Aug con.close() except Exception as es: messagebox.showerror("Error",f"Error Due to: {str(es)}",parent=self.root)

#==========function to clear the fields after success=============== def clear(self): self.email.delete(0, END)

self.passw.delete(0, END)

# ====Function to delete current page and jump on Read More Page===== def read\_More\_page(self):

self.root.destroy()

import Read\_More

root=Tk() obj=Register(root)

root.mainloop()

##### User DashBoard

#----------!!!\_This is the main window where main operation are going to perform\_!!!--------------------

------

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#====================================Importing required

libraries=============================================

#-------------------------------------------------------------------------------------------------------------

from tkinter.ttk import Label, Button

from tkinter import StringVar, Checkbutton, Tk, Label, Button, messagebox from PIL import Image,ImageTk from tkinter import filedialog from tkinter import simpledialog

import random import cv2 import numpy as np

import os #\_\_\_Note:- {This Library is used to create folder}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#===================================Creating class========================================================

#---------------------------------------------------------------------------------------------------------

class Register:

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#=======creation function to wrape all task===============

#------------------------------------------------------------------------------------------------------------ def \_\_init\_\_(self, root):

self.root = root self.root.title("ConsoleLancer")

self.root.geometry("1600x750+0+0")

#=======================================================================

=================================

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_BACK END

CODE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#---------------------This Code is inderectly cennected with front end code------------------------------

#======================================================================= =================================

#==========Creating function for taking sample number from user=================== def Sample\_Number():

global Sample\_Number\_R

USER\_INP = simpledialog.askstring(title="Test", prompt="Please Enter The Number of Sample You want")

Temp\_value=USER\_INP

Sample\_Number\_R = int(Temp\_value)

#============Creating Variable for Checking Condition of Download button================

self.file = 0 self.path = 0 self.filter = 0

#-----------------Variable For Checking Default Displaye Image ------------ self.sample\_Image = 0

#======================Creating function to upload file=================================== def Upload\_file(): self.filename = 0

self.filename = filedialog.askopenfilename(initialdir='/guis', title="Open An Image File", filetypes=(("PNG File", "\*.png"), ("All Files", "\*.\*"))) #----------Passing Filename Address in sel.file variable for if condition ----------------- self.file = self.filename self.sample\_Image = self.filename

self.filename1 = self.filename2 = self.filename

#-----Resizing Sample image in 700x270------------- my\_image = Image.open(self.filename1)

resized = my\_image.resize((700, 270), Image.ANTIALIAS) self.my\_image1 = ImageTk.PhotoImage(resized)

#---------Resizing Image in 220x220---------------- my\_image1 = Image.open(self.filename2)

resized1 = my\_image1.resize((220, 220), Image.ANTIALIAS) self.my\_image2 = ImageTk.PhotoImage(resized1)

self.Filter\_user\_Sample = Label(root, image=self.my\_image2)

# ---------Placing Sample Image as Default view-----

self.default\_User\_sample = Label(root, image=self.my\_image1)

if self.filename == 0:

self.default\_User\_sample.place(x=310, y=280)

else:

cv2.imread(Hide\_the\_chk\_buttons()) self.default\_User\_sample.place\_forget() self.default\_User\_sample.place(x=310, y=280)

# ------Asking user for number of sample-----------

cv2.imread(Sample\_Number()) # --Asking user for number of sample

#==============Creating function for Setting Path================= def savefile(): global filepath

filepath = filedialog.askdirectory()

self.path = filepath

#=============Creating Function to create new folder============== def createFolder(directory): # Creating function to create New Folder try: if not os.path.exists(directory): os.makedirs(directory) except OSError: print('Error: Creating directory. ' + directory)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_

#=======Filter:- Creating Filters To Generate Varius Sample of Data===================

#-------------------------------------------------------------------------------------

#=========Creating Function for Resize Filter============ def Resize\_filter():

global resize\_Sample\_number resize\_Sample\_number = Sample\_Number\_R if Resize\_variable.get() == "Resize":

Sample\_folder = createFolder('%s/Resize\_Effect\_Sample/' % (filepath)) #Calling function to create Folder

import cv2

for resize\_Loop in range(0, resize\_Sample\_number): # Creating loop img = cv2.imread(self.filename) # calling user input image w = random.randint(80, 1000) # passing random value for ramdom width h = random.randint(80, 1000) # passing random value for ramdom width

width, height = w, h # Passing x and y in height and width

imageresize = cv2.resize(img, (width, height)) # passing Loop width and height image in variable

cv2.imwrite('%s/Resize\_Effect\_Sample/%s.jpg' % (filepath, resize\_Loop+1), imageresize) # Saving image at specific path

# ----Note:- This code is working properly--------------

else: pass

# ============Invert Filter ==================== def Invert\_filter(): if Invert\_variable.get() == "invert": import cv2

global invert\_Sample\_number

invert\_Sample\_number = Sample\_Number\_R

Sample\_folder = createFolder('%s/Invert\_Effect\_Sample/' % (filepath)) # Calling function to create Folder

def invert\_image(): image = cv2.imread(self.filename) image1 = cv2.bitwise\_not(image)

cv2.imwrite('%s/Invert\_Effect\_Sample/0.jpg' % (filepath),image1) # Saving Byte change inverted image

for invert\_loop in range(0,invert\_Sample\_number): channel = random.uniform(0,481) image2 = (channel - image)

cv2.imwrite('%s/Invert\_Effect\_Sample/%s.jpg' % (filepath, invert\_loop+1), image2)

#Saving inverted image generated by random value

cv2.imread(invert\_image())

#--------------Note:- This Code is Working Properly---------------- else: pass

# =========Flip filter=============== def Flip\_filter():

if Flip\_variable.get() == "flip":

Sample\_folder = createFolder('%s/Flip\_Effect\_Sample/' % (filepath)) # Calling function to create Folder

import cv2

originalImage = cv2.imread(self.filename) # Taking Image For Generating Sample flipv = cv2.flip(originalImage, 1) # Generating Sample flipbv = cv2.flip(originalImage, -0) flipbh = cv2.flip(originalImage, -1)

cv2.imwrite('%s/Flip\_Effect\_Sample/1.jpg' % (filepath,), originalImage) # Saving Generated Image

cv2.imwrite('%s/Flip\_Effect\_Sample/2.jpg' % (filepath), flipv) cv2.imwrite('%s/Flip\_Effect\_Sample/3.jpg' % (filepath), flipbv) cv2.imwrite('%s/Flip\_Effect\_Sample/4.jpg' % (filepath), flipbh) else: pass

# --------------Note:- This Code is working properly--------------------------

# ===============rotate filter ====================== def Rotate\_filter():

if Rotate\_variable.get() == "rotate":

Sample\_folder = createFolder('%s/Rotate\_Effect\_Sample/' % (filepath)) # Calling function to create Folder import cv2

originalImage = cv2.imread(self.filename) # Taking Image For Generating Sample

img\_rotate = cv2.rotate(originalImage, cv2.ROTATE\_90\_CLOCKWISE) # Generating Sample

img\_rotate90 = cv2.rotate(originalImage, cv2.ROTATE\_90\_COUNTERCLOCKWISE) img\_rotate180 = cv2.rotate(originalImage, cv2.ROTATE\_180)

cv2.imwrite('%s/Rotate\_Effect\_Sample/1.jpg' % (filepath), originalImage) # Saving Generated Sample

cv2.imwrite('%s/Rotate\_Effect\_Sample/2.jpg' % (filepath), img\_rotate) cv2.imwrite('%s/Rotate\_Effect\_Sample/3.jpg' % (filepath), img\_rotate90) cv2.imwrite('%s/Rotate\_Effect\_Sample/4.jpg' % (filepath), img\_rotate180) # ------------------Note:- This filter is working properly--------------------

else: pass

#==========Creating function for Hue Filter============== def Hue\_filter(): if Hue\_variable.get()=="hue":

Sample\_folder = createFolder('%s/Hue\_Effect\_Sample/' % (filepath)) # Calling function to create Folder

global hue\_Sample\_number hue\_Sample\_number = Sample\_Number\_R def hue\_image():

image = cv2.imread(self.filename) #Taking Sample

for name in range(1, hue\_Sample\_number + 1): saturation = random.randint(5,5001) #Passing Random number for Diffrent Sample hue\_Efec = random.randint(10, 1000)

image = cv2.cvtColor(image, cv2.COLOR\_BGR2HSV) v = image[:,:,2]

v = np.where(v <= hue\_Efec + saturation, v - saturation, hue\_Efec) image[:, :, 2] = v

image = cv2.cvtColor(image, cv2.COLOR\_HSV2BGR)

cv2.imwrite('%s/Hue\_Effect\_Sample/%s.jpg' % (filepath,name1), image)

for name1 in range(1,hue\_Sample\_number+1): hue\_image()

#----------------Note:- This Code Is Working Properly----------------------

else: pass

#=================Creating function for Light Filter======================= def Light\_filter(): if Light\_filter\_variable.get() == "light":

Sample\_folder = createFolder('%s/Light\_Effect\_Sample/' % (filepath)) # Calling function to create Folder

global light\_Filter\_Sample\_number

light\_Filter\_Sample\_number = Sample\_Number\_R import cv2 import numpy as np def add\_light():

image = cv2.imread(self.filename) # Taking Sample

#for name in range(1, light\_Filter\_Sample\_number + 1):

gamma = random.uniform(-90,90) # Passing Random number for Diffrent Sample if gamma==0: gamma=gamma+1 invGamma = 1.0 / gamma

table = np.array([((i / 255.0) \*\* invGamma) \* 255 for i in np.arange(0.5, 256)]).astype("uint8")

image1 = cv2.LUT(image, table) if gamma >= 1:

cv2.imwrite('%s/Light\_Effect\_Sample/%s.jpg' % (filepath,name), image1) else:

cv2.imwrite('%s/Light\_Effect\_Sample/%s.jpg' % (filepath,name), image1) for name in range(1, light\_Filter\_Sample\_number + 1):

add\_light()

#--------------------Note:- This Code is working properly----------------- else: pass

#============Creating Function for Light Color Filter================== def Light\_color\_filter(): if Light\_color\_filters\_variable.get() == "lColor":

Sample\_folder = createFolder('%s/Light\_color\_Effect\_Sample/' % (filepath)) # Calling function to create Folder

global light\_Color\_Sample\_number

light\_Color\_Sample\_number = Sample\_Number\_R import cv2

import numpy as np

def add\_light\_color():

image = cv2.imread(self.filename) # Taking Sample gamma = random.uniform(0.1, 2.1) # Passing Random number for Diffrent Sample color = random.randint(50, 250) # Passing Random number for Diffrent Sample invGamma = 1.0 / gamma image = (color - image) table = np.array([((i / 255.0) \*\* invGamma) \* 255 for i in np.arange(0, 256)]).astype("uint8")

image = cv2.LUT(image, table) if gamma >= 1: cv2.imwrite('%s/Light\_color\_Effect\_Sample/%s.jpg' % (filepath, name),image)

else:

cv2.imwrite('%s/Light\_color\_Effect\_Sample/%s.jpg' % (filepath, name), image)

for name in range(1, light\_Color\_Sample\_number + 1):

add\_light\_color()

#-------------Note:- This Code is working properly----------------------------- else: pass

#===========Creating Fucntion for Seturation Filter================ def Seturate\_filter(): if Seturate\_variable.get()=="Seturate\_Image":

Sample\_folder = createFolder('%s/Seturate\_Effect\_Sample/' % (filepath)) # Calling function to create Folder

global Seturation\_Sample\_number

Seturation\_Sample\_number = Sample\_Number\_R import cv2

import numpy as np def saturation\_image():

#image = cv2.imread(self.filename) # Taking Sample for name in range(1, Seturation\_Sample\_number + 1): image = cv2.imread(self.filename)

saturation = random.randint(5,400) # Passing Random number for Diffrent Sample saturation1 = random.randint(5, 400)

image = cv2.cvtColor(image, cv2.COLOR\_BGR2HSV)

v = image[:, :, 2]

v = np.where(v <= saturation1 - saturation, v + saturation, saturation1) image[:, :, 2] = v

image = cv2.cvtColor(image, cv2.COLOR\_HSV2BGR)

cv2.imwrite('%s/Seturate\_Effect\_Sample/%s.jpg' % (filepath,name), image) cv2.imshow("w",image)

#for name in range(1, Seturation\_Sample\_number + 1):

#cv2.imwrite('%s/Seturate\_Effect\_Sample/%s.jpg' % (filepath, name), image) saturation\_image()

#-----------------Note:- This Code is working properly----------------

#???????????????Note:- But Generating only One Image Need To Work on Loop?????????????????? else: pass

#========Creating Function for Gray Scale Image Filter===================================== def Gray\_scale\_Filter(): if Gray\_scale\_variable.get() == "Gray":

Sample\_folder = createFolder(

'%s/Rectangle\_covered\_Sample/' % (filepath)) # Calling function to create Folder global gray\_Scale\_Sample\_number gray\_Scale\_Sample\_number = Sample\_Number\_R for name in range(1, gray\_Scale\_Sample\_number + 1):

image = cv2.imread(self.filename) height, width = image.shape[:2] height\_value = random.randint(10, 50) width\_value = random.randint(10, 50) position\_x = random.randint(50, height) position\_y = random.randint(50, width) color3 = random.randint(50, 200) color1 = random.randint(50, 200) color2 = random.randint(50, 200)

cv2.rectangle(image, pt1=(position\_y, position\_x), pt2=(height\_value, width\_value), color=(color1, color2, color3), thickness=-1)

cv2.imwrite('%s/Rectangle\_covered\_Sample/%s.jpg' % (filepath, name), image)

# ???????????Note:- Generating only one sample Work on it????????????????? else: pass

#===========================Creating Function for Addepive Filter============================= def Addeptive\_gaussian\_filter(): if Addeptive\_variable.get()=="addept":

Sample\_folder = createFolder(

'%s/Addeptive\_Effect\_Sample/' % (filepath)) # Calling function to create Folder import cv2

global Addeptive\_Sample\_number

Addeptive\_Sample\_number = Sample\_Number\_R

def addeptive\_gaussian\_noise(): image = cv2.imread(self.filename) # Taking Sample

Addept\_diffs = random.randint(100, 300) # Passing Random number for Diffrent Sample:

Addept\_diffh = random.randint(100, 300) # Passing Random number for Diffrent Sample:

Addept\_diffv = random.randint(100, 300) # Passing Random number for Diffrent Sample:

h, s, v = cv2.split(image)

s = cv2.adaptiveThreshold(s, Addept\_diffs, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C, cv2.THRESH\_BINARY\_INV, 11, 2)

h = cv2.adaptiveThreshold(h,Addept\_diffh, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C, cv2.THRESH\_BINARY\_INV, 11, 2)

v = cv2.adaptiveThreshold(v, Addept\_diffv, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C, cv2.THRESH\_BINARY\_INV, 11, 2)

image = cv2.merge([h, s, v])

cv2.imshow("w", image)

cv2.imwrite('%s/Addeptive\_Effect\_Sample/%s.jpg' % (filepath, name), image)

for name in range(1, Addeptive\_Sample\_number + 1):

addeptive\_gaussian\_noise()

#???????????????This Code Is Not Generating any sample But Not Throwing Error As Well???????????????? else: pass

#=======================Creating Function for Contrass Filter============================ def Contrass\_filter(): if Contrass\_variable.get()=="Contra":

Sample\_folder = createFolder('%s/Contrass\_Effect\_Sample/' % (filepath)) # Calling function to create Folder import cv2

global contrass\_Sample\_number

contrass\_Sample\_number = Sample\_Number\_R

def contrast\_image():

for name in range(1, contrass\_Sample\_number + 1): image = cv2.imread(self.filename) # Taking Sample contrast = random.uniform(-150,199) # Passing Random number for Diffrent Sample: image = cv2.cvtColor(image, cv2.COLOR\_BGR2HSV) image[:, :, 2] = [

[max(pixel - contrast, 0) if pixel < 190 else min(pixel + contrast, 255) for pixel in row] for

row in image[:, :, 2]]

image = cv2.cvtColor(image, cv2.COLOR\_HSV2BGR)

cv2.imwrite('%s/Contrass\_Effect\_Sample/%s.jpg' % (filepath, name), image)

contrast\_image()

#????????Note:- This Code Is Taking Too Much Time And Genrating Only one Image But Workin With out Any Syntax Error????? else: pass

#=====================Creating Function for Edge Canny Filter========================= def Edge\_canny\_filter():

if Edge\_detect\_variable.get()=="cany":

Sample\_folder = createFolder('%s/Edge\_Canny\_Effect\_Sample/' % (filepath)) # Calling function to create Folder import cv2

global edge\_cany\_Sample\_number edge\_cany\_Sample\_number = Sample\_Number\_R def edge\_detect\_canny\_image():

for name in range(1, edge\_cany\_Sample\_number + 1): image = cv2.imread(self.filename) # Taking Sample cany\_diff1 = random.randint(0, 100) # Passing Random number for Diffrent Sample: cany\_diff2 = random.randint(0, 100) # Passing Random number for Diffrent Sample: image = cv2.Canny(image,cany\_diff1,cany\_diff2)

cv2.imwrite('%s/Edge\_Canny\_Effect\_Sample/%s.jpg' % (filepath, name), image)

edge\_detect\_canny\_image() else: pass

#===========Creating Function for Transformation Filter================== def Transformation\_filter(): if Transformation\_variable.get()=="Transfom":

Sample\_folder = createFolder(

'%s/Transform\_Effect\_Sample/' % (filepath)) # Calling function to create Folder import cv2

import numpy as np

global transformation\_Sample\_number

transformation\_Sample\_number = Sample\_Number\_R

def transformation\_image():

for name in range(1, transformation\_Sample\_number + 1):

image = cv2.imread(self.filename) rows, cols, ch = image.shape ptsx1 = random.randint(0, 500) ptsx2 = random.randint(0, 500) pts1 = np.float32([[ptsx1, ptsx2], [200, 50], [50, 200]]) pts2 = np.float32([[10, 100], [200, 50], [100, 250]]) M = cv2.getAffineTransform(pts1, pts2) image = cv2.warpAffine(image, M, (cols, rows))

cv2.imwrite('%s/Transform\_Effect\_Sample/%s.jpg' % (filepath, name), image)

transformation\_image() else: pass

#=============Creating Function for Embossed Filter==================== def crop(): if Emboss\_variable.get()=="embs":

Sample\_folder = createFolder('%s/Crop\_Sample/' % (filepath)) # Calling function to create

Folder

import cv2 import numpy as np global crop\_Sample\_number

crop\_Sample\_number = Sample\_Number\_R

for name in range(1,crop\_Sample\_number + 1):

image = cv2.imread(self.filename) '''x = random.uniform(.01, .99) y = random.uniform(.01, .99)''' x = random.uniform(.01, .50) y = random.uniform(.60, .99)

height, width = image.shape[:2]

start\_row, start\_col = int(height \* x), int(width \* x)

end\_row, end\_col = int(height \* y), int(width \* y)

cropped = image[start\_row:end\_row, start\_col:end\_col] cv2.imwrite('%s/Crop\_Sample/%s.jpg' % (filepath, name),cropped) else: pass

#================Creating Function for Translation Filter================================= def Translation\_filter(): if Translation\_variable.get()=="Translation":

Sample\_folder = createFolder(

'%s/Translation\_Effect\_Sample/' % (filepath)) # Calling function to create Folder import cv2 import numpy as np

global translation\_Sample\_number

translation\_Sample\_number = Sample\_Number\_R

def translation\_image():

image = cv2.imread(self.filename) # Taking Sample translation\_diff1 = random.uniform(-150,150) # Passing Random number for Diffrent Sample: translation\_diff2 = random.uniform(-150,150) # Passing Random number for Diffrent Sample:

rows, cols, c = image.shape

M = np.float32([[1,0,translation\_diff1], [0, 1,translation\_diff2]]) image = cv2.warpAffine(image, M, (cols, rows))

cv2.imwrite('%s/Translation\_Effect\_Sample/%s.jpg' % (filepath, name), image)

for name in range(1,translation\_Sample\_number + 1):

translation\_image() else: pass

#==========creating function for salt Filter========================

def Salt\_filter(): if salt\_and\_paper\_variable.get()=="Salt\_paper":

Sample\_folder = createFolder(

'%s/Salt\_Effect\_Sample/' % (filepath)) # Calling function to create Folder global edge\_cany\_Sample\_number

edge\_cany\_Sample\_number = Sample\_Number\_R import numpy as np

import cv2

for name in range(1, edge\_cany\_Sample\_number + 1): image = cv2.imread(self.filename) # Taking Sample color3 = random.randint(50, 200) color1 = random.randint(50, 200) color2 = random.randint(50, 200) height, width = image.shape[:2] radius\_value = random.randint(10, 50) position\_circle = random.randint(50, height) position\_circle = random.randint(50, width)

cv2.circle(image, center=(position\_circle, position\_circle), radius=radius\_value, color=(color1, color2, color3), thickness=-10) cv2.imwrite('%s/Salt\_Effect\_Sample/%s.jpg' % (filepath, name), image) else: pass

#=============Creating Function for Sharp Filter================== def Sharp\_filter(): if Sharp\_variable.get()=="Sharp\_value": global edge\_cany\_Sample\_number

edge\_cany\_Sample\_number = Sample\_Number\_R

Sample\_folder = createFolder(

'%s/Pencil\_Shade\_Sample/' % (filepath)) # Calling function to create Folder import cv2

#import numpy as np

import random

def sharpen\_image(): for name in range(1, edge\_cany\_Sample\_number + 1): #image = cv2.imread(self.filename) # Taking Sample color\_image = cv2.imread(self.filename) sm = random.randint(1, 150) sr = random.uniform(0.009, 0.9)

cartoon\_image1, bawla = cv2.pencilSketch(color\_image, sigma\_s=sm, sigma\_r=sr, shade\_factor=0.02)

cv2.imshow('cartoon', cartoon\_image1)

cv2.imwrite('%s/Pencil\_Shade\_Sample/%s.jpg' % (filepath, name), cartoon\_image1) sharpen\_image()

# ???????????Note:- Generating only one sample?????????????????

else: pass

#=========Creating function for Dilation Filter================ def Dilation\_filter(): if dilation\_variable.get()=="dilation\_value": global dela\_cany\_Sample\_number

dela\_cany\_Sample\_number = Sample\_Number\_R

Sample\_folder = createFolder(

'%s/Dilation\_Effect\_Sample/' % (filepath)) # Calling function to create Folder import cv2

import numpy as np

def dilation\_image(): for name in range(1, dela\_cany\_Sample\_number + 1): image = cv2.imread(self.filename) # Taking Sample dila\_diff1 = random.randint(0,51) # Passing Random number for Diffrent Sample: dila\_diff2 = random.randint(0,51) # Passing Random number for Diffrent Sample:

kernel = np.ones((dila\_diff1, dila\_diff2), np.uint8) image = cv2.dilate(image, kernel, iterations=1)

cv2.imwrite('%s/Dilation\_Effect\_Sample/%s.jpg' % (filepath, name), image)

dilation\_image()

# ???????????Note:- Generating only one sample?????????????????

else: pass

#==================Creating function for Blure Filter=================== def Blure\_filter(): if Blure\_variable.get()=="Blure\_value": global dela\_cany\_Sample\_number

dela\_cany\_Sample\_number = Sample\_Number\_R

Sample\_folder = createFolder(

'%s/Blure\_Effect\_Sample/' % (filepath)) # Calling function to create Folder import cv2

def averageing\_blur(): for name in range(1, dela\_cany\_Sample\_number + 1): image = cv2.imread(self.filename) # Taking Sample avgBlur\_diff1 = random.randint(1,41) # Passing Random number for Diffrent Sample: avgBlur\_diff2 = random.randint(1, 41)

image = cv2.blur(image, (avgBlur\_diff1, avgBlur\_diff2))

cv2.imwrite('%s/Blure\_Effect\_Sample/%s.jpg' % (filepath, name), image)

averageing\_blur()

# ???????????Note:- Generating only one sample????????????????? else: pass

#============Creating Function for Black Hat Filter=================== def cartoon(): if Black\_hat\_variable.get()=="Black\_hat\_value":

Sample\_folder = createFolder(

'%s/Black\_Hat\_Effect\_Sample/' % (filepath)) # Calling function to create Folder import cv2 import numpy as np

global black\_Hat\_Sample\_number

black\_Hat\_Sample\_number = Sample\_Number\_R

for name in range(1, black\_Hat\_Sample\_number + 1): image = cv2.imread(self.filename) # Taking Sample sm = random.randint(1,1000) sr = random.uniform(0.001,1.99)

image1 = cv2.stylization(image, sigma\_s=sm, sigma\_r=sr)

#cv2.imwrite('%s/Test\_Sample/%s.jpg' % (filepath, name), image) cv2.imwrite('%s/Black\_Hat\_Effect\_Sample/%s.jpg' % (filepath, name), image1) # ???????????Note:- Generating only one sample????????????????? else: pass

#=============Creating function for Top Hat Filter================ def Top\_Hat\_filter(): if Top\_hat\_variable.get()=="Top\_hat\_value":

Sample\_folder = createFolder(

'%s/Top\_Hat\_Effect\_Sample/' % (filepath)) # Calling function to create Folder import cv2 import numpy as np global top\_Hat\_Sample\_number

top\_Hat\_Sample\_number = Sample\_Number\_R

for name in range(1, top\_Hat\_Sample\_number + 1): image = cv2.imread(self.filename) # Taking Sample

top\_Hat\_diff1 = random.randint(200, 500) # Passing Random number for Diffrent Sample:

kernel = np.ones((top\_Hat\_diff1, top\_Hat\_diff1), np.uint8) image = cv2.morphologyEx(image, cv2.MORPH\_TOPHAT, kernel)

cv2.imwrite('%s/Top\_Hat\_Effect\_Sample/%s.jpg' % (filepath, name), image)

# ???????????Note:- Generating only one sample????????????????? else: pass

#=========This is a extra filter add for testing porpus=================== def Test\_filter(): if blank\_variable.get()=="test\_value":

Sample\_folder = createFolder(

'%s/Test\_Sample/' % (filepath)) # Calling function to create Folder global top\_Hat\_Sample\_number

top\_Hat\_Sample\_number = Sample\_Number\_R import cv2 for name in range(1, top\_Hat\_Sample\_number + 1): image = cv2.imread(self.filename) # Taking Sample top\_Hat\_diff1 = random.randint(1,10) # Passing Random number for Diffrent Sample: image = cv2.blur(image, (top\_Hat\_diff1, top\_Hat\_diff1)) cv2.imwrite('%s/Test\_Sample/%s.jpg' % (filepath, name), image) cv2.imshow("w", image)

cv2.waitKey(0) else: pass

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_

#==================Creation function for Data Set

Generating=========================

#------------------------------------------------------------------------------------ def download\_Button(): if self.file ==0 or self.path==0: if self.file==0:

messagebox.showwarning("Warning", "Please Upload Semple Image First", parent=self.root) if self.path==0:

messagebox.showwarning("warning", "Please select path first", parent=self.root) else: pass

else:

#---Note:- Callig Function in Select Filter Frame----------- cv2.imread(Resize\_filter()) cv2.imread(Invert\_filter()) cv2.imread(Flip\_filter()) cv2.imread(Rotate\_filter()) cv2.imread(Hue\_filter()) cv2.imread(Light\_filter()) cv2.imread(Light\_color\_filter()) cv2.imread(Seturate\_filter())

cv2.imread(Addeptive\_gaussian\_filter()) cv2.imread(Contrass\_filter()) cv2.imread(Edge\_canny\_filter()) cv2.imread(Transformation\_filter()) cv2.imread(crop()) cv2.imread(Gray\_scale\_Filter()) cv2.imread(Translation\_filter()) cv2.imread(Salt\_filter()) cv2.imread(Sharp\_filter()) cv2.imread((Dilation\_filter())) cv2.imread(Blure\_filter()) cv2.imread(cartoon()) cv2.imread(Top\_Hat\_filter())

#----------------Note:- This Code Is Working Properly----------------

#---------Note:- Calling Function in More Filter Frame ---------------- cv2.imread(Test\_filter())

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_

#===============Creating variable to check on value or off value of check box==============

#---------------Note :- These variable are for Select Filter Frame------------------------- #------------------------------------------------------------------------------------------- Resize\_variable = StringVar()

Flip\_variable = StringVar()

Invert\_variable = StringVar()

Hue\_variable = StringVar()

Rotate\_variable = StringVar()

Light\_filter\_variable = StringVar()

Light\_color\_filters\_variable = StringVar()

Seturate\_variable = StringVar()

Addeptive\_variable = StringVar()

Contrass\_variable = StringVar()

Edge\_detect\_variable = StringVar()

Transformation\_variable = StringVar()

Emboss\_variable = StringVar()

Gray\_scale\_variable = StringVar() Translation\_variable = StringVar() salt\_and\_paper\_variable = StringVar() Sharp\_variable = StringVar() dilation\_variable = StringVar() Blure\_variable = StringVar()

Black\_hat\_variable = StringVar()

Top\_hat\_variable = StringVar()

#------Note:- Above Code is working properly---------- blank\_variable = StringVar()

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_

#-----------------Putting Buttons on Screen------------------------------------------

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_

# \_\_\_\_\_\_\_Show\_Feature:- Function For Putting Button On Screen\_\_\_\_\_\_\_\_\_ def show\_Feature(): upload\_Sample.place(x=290, y=580) save\_Button.place(x=410, y=580) Show\_Filter\_button.place(x=530, y=580) #select\_More.place(x=650, y=580) hide\_button.place(x=650, y=580) Generate\_Sample.place(x=780, y=580) feature\_button1.place(x=900, y=580)

#\_\_\_\_\_\_Hide\_Feature:- Fucntion to Hide Feature Button From Screen\_\_\_\_\_ def hide\_Feature(): upload\_Sample.place\_forget() save\_Button.place\_forget() Show\_Filter\_button.place\_forget() #select\_More.place\_forget() hide\_button.place\_forget() feature\_button1.place\_forget() Generate\_Sample.place\_forget()

def Hide\_the\_chk\_2(): chk\_size.place\_forget() chk\_invert.place\_forget() chk\_crop.place\_forget() chk\_blure.place\_forget() chk\_hue.place\_forget() chk\_light.place\_forget() chk\_light\_color.place\_forget() chk\_setu.place\_forget() chk\_gray.place\_forget() chk\_addeptive.place\_forget() chk\_Contrass.place\_forget() chk\_Edge\_cany.place\_forget() Transfom\_check.place\_forget() chk\_emboss.place\_forget() chk\_Translation.place\_forget() chk\_salt\_paper.place\_forget() chhk\_Sharp.place\_forget() chhk\_dilation.place\_forget() Chk\_Blure.place\_forget()

chhk\_Black\_hat.place\_forget()

#\_\_\_\_\_\_Hide\_Check:- Fucntion for Hiding Check Box\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ def Hide\_the\_chk\_buttons(): chk\_size.place\_forget() chk\_invert.place\_forget() chk\_crop.place\_forget() chk\_blure.place\_forget() chk\_hue.place\_forget() chk\_light.place\_forget() chk\_light\_color.place\_forget() chk\_setu.place\_forget() chk\_gray.place\_forget() chk\_addeptive.place\_forget() chk\_Contrass.place\_forget() chk\_Edge\_cany.place\_forget() Transfom\_check.place\_forget() chk\_emboss.place\_forget() chk\_Translation.place\_forget() chk\_salt\_paper.place\_forget() chhk\_Sharp.place\_forget() chhk\_dilation.place\_forget() Chk\_Blure.place\_forget() chhk\_Black\_hat.place\_forget()

Chk\_Test.place\_forget()

# ---------Checking User input image exist or not ------------- if self.sample\_Image == 0: self.defaultImage2.place\_forget() self.defaultImage.place(x=310, y=280) else:

# -----Placing 700x270 image on screen------ self.Filter\_user\_Sample.place\_forget() self.default\_User\_sample.place(x=310, y=280)

Check\_blank.place\_forget()

#\_\_\_\_\_\_\_More\_Filter:- Function for putting some extra filter on screen def more\_Filter(): Check\_blank.deselect()

Check\_blank.place(x=600, y=280)

chk\_size.place\_forget() chk\_invert.place\_forget() chk\_crop.place\_forget() chk\_blure.place\_forget() chk\_hue.place\_forget() chk\_light.place\_forget() chk\_light\_color.place\_forget() chk\_setu.place\_forget() chk\_gray.place\_forget() chk\_addeptive.place\_forget() chk\_Contrass.place\_forget() chk\_Edge\_cany.place\_forget() Transfom\_check.place\_forget() chk\_emboss.place\_forget() chk\_Translation.place\_forget() chk\_salt\_paper.place\_forget() chhk\_Sharp.place\_forget() chhk\_dilation.place\_forget() Chk\_Blure.place\_forget() chhk\_Black\_hat.place\_forget()

Chk\_Test.place\_forget()

# ---------Checking User input image exist or not ------------- if self.sample\_Image == 0: self.defaultImage.place\_forget() self.defaultImage2.place(x=310, y=280) else:

self.default\_User\_sample.place\_forget() self.defaultImage.place\_forget()

self.Filter\_user\_Sample.place(x=310, y=280)

#\_\_\_\_\_\_\_\_\_Filter\_Show:-Function For Putting CHeck Button On Screen\_\_\_\_\_\_\_\_\_ def Filter\_show():

# ----------Creating Resized Filter Check Box -------------------- 1 chk\_size.deselect()

chk\_size.place(x=600, y=280)

# ----------Creating Invert Filter Check Box -------------------- 2 chk\_invert.deselect()

chk\_invert.place(x=600, y=315)

# ----------Creating flip Filter Check Box -------------------- 3 chk\_crop.deselect()

chk\_crop.place(x=600, y=350)

# ----------Creating Rotate Filter Check Box -------------------- 4 chk\_blure.deselect()

chk\_blure.place(x=600, y=385)

# ----------Creating Hue Filter Check Box -------------------- 5 chk\_hue.deselect()

chk\_hue.place(x=600, y=420)

# ----------Creating ligth Filter Check Box -------------------- 6 chk\_light.deselect()

chk\_light.place(x=600, y=455)

# ----------Creating ligth color Filter Check Box -------------------- 7 chk\_light\_color.deselect()

chk\_light\_color.place(x=600, y=490)

# ----------Creating Seturation Filter Image Filter Check Box -------------------- 8 chk\_setu.deselect()

chk\_setu.place(x=770, y=280)

# ----------Creating Gray Scale Filter Check Box -------------------- 9 chk\_gray.deselect()

chk\_gray.place(x=770, y=315)

# ----------Creating Adeptive Gaussian Check Box -------------------- 10 chk\_addeptive.deselect()

chk\_addeptive.place(x=770, y=350)

# ----------Creating Contrass Check Box -------------------- 11 chk\_Contrass.deselect()

chk\_Contrass.place(x=770, y=385)

# ----------Creating Edge Detect Canny Check Box -------------------- 12 chk\_Edge\_cany.deselect() chk\_Edge\_cany.place(x=770, y=420)

# ----------Creating Transformation Check Box -------------------- 13

Transfom\_check.deselect()

Transfom\_check.place(x=770, y=455)

# ----------Creating Emboss Check Box -------------------- 14 chk\_emboss.deselect() chk\_emboss.place(x=770, y=490)

# ----------Creating Translation Filter Check Box -------------------- 15 chk\_Translation.deselect() chk\_Translation.place(x=940, y=280)

# ----------Creating Salt And Paper Check Box -------------------- 16 chk\_salt\_paper.deselect() chk\_salt\_paper.place(x=940, y=315)

# ----------Creating Sharp Check Box -------------------- 17 chhk\_Sharp.deselect() chhk\_Sharp.place(x=940, y=350)

# ----------Creating Blank Check Box -------------------- 18 chhk\_dilation.deselect() chhk\_dilation.place(x=940, y=385)

# ----------Creating Blank Check Box -------------------- 19

Chk\_Blure.deselect()

Chk\_Blure.place(x=940, y=420)

# ----------Creating Blank Check Box -------------------- 20 chhk\_Black\_hat.deselect() chhk\_Black\_hat.place(x=940, y=455)

# ----------Creating Blank Check Box -------------------- 21

Chk\_Test.deselect()

Chk\_Test.place(x=940, y=490)

# ---------Checking User input image exist or not ------------- if self.sample\_Image == 0:

self.defaultImage.place\_forget() self.defaultImage2.place(x=310, y=280) else:

self.default\_User\_sample.place\_forget() self.defaultImage.place\_forget() self.Filter\_user\_Sample.place(x=310, y=280)

#----------Removing Blank Check Box------------

Check\_blank.place\_forget()

#=======================================================================

=======================================

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_FRONT END

CODE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#----------------------Front end code in written here but packed in back end code ---------------------------

--

#=======================================================================

=======================================

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# ================Main-Background=============================

# ------------------------------------------------------------ self.bg = ImageTk.PhotoImage(file="bg.png")

main\_Background = Label(self.root, image=self.bg).place(x=0, y=0, relwidth=1, relheight=1)

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# ================Sub-Background=============================

# ----------------------------------------------------------- self.left = ImageTk.PhotoImage(file="aug1.png")

left = Label(self.root, image=self.left).place(x=220, y=130, width=900, height=500)

#------------------------------------------------------------------------

#-------------------------Creating CheckBox -----------------------------

#------------------------------------------------------------------------ # ----------Creating Resized Filter Check Box -------------------- 1

chk\_size = Checkbutton(left, text="Resize", variable=Resize\_variable,bg="#4e4e4e", onvalue="Resize", offvalue=0,

font=("time new roman", 12))

# ----------Creating Invert Filter Check Box -------------------- 2

chk\_invert = Checkbutton(left, text="invert", variable=Invert\_variable, onvalue="invert", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating flip Filter Check Box -------------------- 3

chk\_crop = Checkbutton(left, text="Flip", variable=Flip\_variable, onvalue="flip", offvalue=0, bg="#4e4e4e", font=("time new roman", 12))

# ----------Creating Rotate Filter Check Box -------------------- 4

chk\_blure = Checkbutton(left, text="Rotate", variable=Rotate\_variable, onvalue="rotate", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Hue Filter Check Box -------------------- 5

chk\_hue = Checkbutton(left, text="Hue", variable=Hue\_variable, onvalue="hue", offvalue=0, bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating ligth Filter Check Box -------------------- 6

chk\_light = Checkbutton(left, text="Light", variable=Light\_filter\_variable, onvalue="light", offvalue=0, bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating ligth color Filter Check Box -------------------- 7

chk\_light\_color = Checkbutton(left, text="Light Color", variable=Light\_color\_filters\_variable, onvalue="lColor", offvalue=0, bg="#4e4e4e", font=("time new roman", 12))

# ----------Creating Seturation Filter Image Filter Check Box -------------------- 8 chk\_setu = Checkbutton(left, text="Seturation", variable=Seturate\_variable, onvalue="Seturate\_Image",

offvalue=0, bg="#4e4e4e", font=("time new roman", 12))

# ----------Creating Adeptive Gaussian Check Box -------------------- 10

chk\_addeptive = Checkbutton(left, text="Addeptive\_gaussian", variable=Addeptive\_variable, onvalue="addept",

offvalue=0, bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Gray Scale Filter Check Box -------------------- 9 chk\_gray = Checkbutton(left, text="Gray Scale", variable=Gray\_scale\_variable, onvalue="Gray", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Contrass Check Box -------------------- 11

chk\_Contrass = Checkbutton(left, text="Contrass", variable=Contrass\_variable, onvalue="Contra", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Edge Detect Canny Check Box -------------------- 12

chk\_Edge\_cany = Checkbutton(left, text="Edge Canny", variable=Edge\_detect\_variable, onvalue="cany", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Transformation Check Box -------------------- 13

Transfom\_check = Checkbutton(left, text="Transformation", variable=Transformation\_variable, onvalue="Transfom", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Emboss Check Box -------------------- 14

chk\_emboss = Checkbutton(left, text="Crop", variable=Emboss\_variable, onvalue="embs", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Translation Filter Check Box -------------------- 15

chk\_Translation = Checkbutton(left, text="Translation", variable=Translation\_variable, onvalue="Translation", offvalue=0,

bg="#4e4e4e", font=("time new roman", 12))

# ----------Creating Salt And Paper Check Box -------------------- 16

chk\_salt\_paper = Checkbutton(left, text="Salt\_And Paper", variable=salt\_and\_paper\_variable, onvalue="Salt\_paper", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Sharp Check Box -------------------- 17

chhk\_Sharp = Checkbutton(left, text="Sharp", variable=Sharp\_variable, onvalue="Sharp\_value", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Blank Check Box -------------------- 18

chhk\_dilation = Checkbutton(left, text="Dilation", variable=dilation\_variable, onvalue="dilation\_value",

offvalue=0, bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Blank Check Box -------------------- 19

Chk\_Blure = Checkbutton(left, text="Blure", variable=Blure\_variable, onvalue="Blure\_value", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Blank Check Box -------------------- 20

chhk\_Black\_hat = Checkbutton(left, text="Black Hat", variable=Black\_hat\_variable, onvalue="Black\_hat\_value",

offvalue=0, bg="#4e4e4e",

font=("time new roman", 12))

# ----------Creating Blank Check Box -------------------- 21

Chk\_Test = Checkbutton(left, text="Top\_Hat", variable=Top\_hat\_variable, onvalue="Top\_hat\_value", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

#---------------Some Extra Filter------------------------------------- # ----------Creating Blank Check Box --------------------

Check\_blank = Checkbutton(left,text="Test", variable=blank\_variable, onvalue="test\_value", offvalue=0,

bg="#4e4e4e",

font=("time new roman", 12))

'''# ----------Creating All Filter Check Box --------------------

chk\_all = Checkbutton(left, text="All Filter", onvalue=1, offvalue=0, bg="#4e4e4e", font=("time new roman", 12)) chk\_all.deselect()

chk\_all.place(x=600, y=455)'''

#--------------------------------------------------------------------------------- #-----------------------Creating Feature Button-----------------------------------

#---------------------------------------------------------------------------------

#\_\_\_\_\_\_\_Default\_Image:- Importing image to show as defaul\_\_\_\_\_\_\_\_\_\_\_

self.result = ImageTk.PhotoImage(file="default\_image.jpeg")

self.defaultImage = Label(self.root, image=self.result, bd=1, bg="#4e4e4e", cursor="hand2")

#---------Checking User input image exist or not ------------- if self.sample\_Image == 0: self.defaultImage.place(x=310, y=280) else: pass

#\_\_\_\_\_\_\_Default\_Image :- For Side View in 220x220\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

self.result2 = ImageTk.PhotoImage(file="default\_image2.jpeg")

self.defaultImage2 = Label(self.root, image=self.result2, bd=1, bg="#4e4e4e", cursor="hand2")

#\_\_\_\_\_\_Show\_Feature:- \_\_\_\_\_Creating Button To Display Feature Button\_\_\_\_\_\_\_\_\_\_\_\_ self.feature\_Image = ImageTk.PhotoImage(file="Show.png")

feature\_button = Button(self.root,activebackground="#4e4e4e", image=self.feature\_Image, borderwidth=0, bg="#4e4e4e", command=show\_Feature) feature\_button.place(x=250,y=580)

#\_\_\_\_\_\_Hide\_Feature:-\_\_\_Creating Button To Hide Feature Button\_\_\_\_\_\_\_\_\_\_\_ self.feature\_Image1 = ImageTk.PhotoImage(file="Hide.png")

feature\_button1 = Button(self.root, image=self.feature\_Image1,activebackground="#4e4e4e", borderwidth=0, bg="#4e4e4e", command=hide\_Feature)

# \_\_\_\_\_\_Upload:- creating button to upload sample image\_\_\_\_\_\_\_\_\_\_\_\_\_ self.download = ImageTk.PhotoImage(file="Upload.png")

upload\_Sample= Button(self.root, image=self.download,activebackground="#4e4e4e", borderwidth=0 ,bg="#4e4e4e",command=Upload\_file)

# \_\_\_\_\_\_Set-Path:- creating button to set Path for saving data set\_\_\_\_\_\_\_\_\_\_ self.select\_path = ImageTk.PhotoImage(file="Select\_path.png")

save\_Button = Button(self.root,image=self.select\_path,activebackground="#4e4e4e", borderwidth=0,bg="#4e4e4e",command=savefile)

# \_\_\_\_\_\_\_\_\_\_ SelectFilter:- \_button is using to show check button Of filter\_\_\_\_\_

self.select\_filter = ImageTk.PhotoImage(file="Select\_Filter.png") Show\_Filter\_button = Button(self.root,

image=self.select\_filter,borderwidth=0,activebackground="#4e4e4e", bg="#4e4e4e", command=Filter\_show)

# \_\_\_\_\_\_\_\_\_\_Select\_More:- button is using to get more filters \_\_\_\_\_\_

'''self.more2= ImageTk.PhotoImage(file="More\_Filter.png")

select\_More = Button(self.root, image=self.more2, bg="#4e4e4e", borderwidth=0, command=more\_Filter)

#????????????????This line May generating error???????????????'''

# \_\_\_\_\_ Hide\_Filter:- butten is using to hide the check buttons \_\_\_\_\_ self.more1 = ImageTk.PhotoImage(file="Hide\_Filter.png")

hide\_button = Button(self.root, image=self.more1, bg="#4e4e4e",activebackground="#4e4e4e", borderwidth=0, command=Hide\_the\_chk\_buttons)

# \_\_\_\_\_Download:- Creating button to generate data set==============================

self.generate = ImageTk.PhotoImage(file="Download.png")

Generate\_Sample = Button(self.root, image=self.generate, bg="#4e4e4e",

activebackground="#4e4e4e",borderwidth=0, cursor="hand2", command=download\_Button)

root=Tk() obj=Register(root)

root.mainloop()

##### Read More

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-------------------------------------------Importing Required Package(API)--------------------------------------

#----------------------------------------------------------------------------------------------------------------

from PIL import ImageTk import tkinter as tk from tkinter import ttk

from tkinter import scrolledtext, END

class Register: #

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -----------------------------------------------------FRONT END CODE---------------------------------------------

# ----------------------------------------------------------------------------------------------------------------

#=====Root Function=========== def \_\_init\_\_(self,root): self.root=root self.root.title("ConsoleLancer") self.root.geometry("1350x740+0+0")

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -------------------------Frame And Background-------------------

# ----------------------------------------------------------------

#

===========================Frames===============================

#-------------------First Frame----------------------------------- self.left = ImageTk.PhotoImage(file="Augmentation.png")

left = tk.Label(self.root, image=self.left)

left.place(x=0, y=0, width=1350, height=195)

#-------------------Second Frame---------------------------------- frame2 = tk.Frame(self.root, bd=2, bg="#111d20") frame2.place(x=0, y=195, width=300, height=547)

#-------------------Third Frame---------------------------------- frame3 = tk.Frame(self.root,bg="#eeeef0") frame3.place(x=300, y=195,width=1050, height=547)

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# -----------------------Buttons----------------------------------- # -----------------------------------------------------------------

#=====================Frame

1======================================

'''This frame is containing Header. Which is already decleared & initialized in above code'''

#=====================Button Area [Frame

2]========================

#==========This button will throw you on user login page===========

self.User\_Login = ImageTk.PhotoImage(file="User.png")

User =

tk.Button(frame2,image=self.User\_Login,activebackground="#111d20",font=("time new roman", 20, "bold"),command=self.user\_page,bd=0,bg="#111d20",fg="#eeeef0", cursor="hand2")

User.place(x=5,y=100)

#==========This page will throw you on Admin login page===========

self.Admin\_Login = ImageTk.PhotoImage(file="Admin.png") Admin =

tk.Button(frame2,image=self.Admin\_Login,activebackground="#111d20",font=("time new roman", 20, "bold"),command=self.admin\_page ,bd=0,bg="#111d20",fg="#eeeef0", cursor="hand2")

Admin.place(x=5,y=200)

#====================Description Area [Frame 3]==================== ttk.Label(frame3,

text="DESCRIPTION",

font=("time new roman", 20, "bold"),

background='#eeeef0',

foreground="gray").place(x=400,y=50)

#=================Creating scrolled Area===========================

text\_area = scrolledtext.ScrolledText(frame3,

wrap=tk.WORD,

width=80,

height=16,

font=("Times New Roman",

15))

text\_area.place(x=100,y=100)

#========Inserting Product Description In Text Area===============

file = open("product\_Description.txt","r") # Reading Product Description from file

for line in file:

x = line # Passing Each line in x to insert in in text area text\_area.insert(END,x) # Inserting Each Line in text area

# Placing cursor in the text area text\_area.focus()

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#-----------------------------------------------BACK END CODE------------------------------------------------------

#------------------------------------------------------------------------------------------------------------------

#=========Function to jump on Admin login page============== def admin\_page(self): self.root.destroy()

import Admin\_login

#=========Function to jump on User login page=============== def user\_page(self): self.root.destroy()

import User\_Login

root= tk.Tk() obj=Register(root) root.resizable(False, False)

root.mainloop()