Python Development

Internship

2020-21

ConsoleLancer

Internship Project Report

ON

Data Augmentation

Session :- 2020-21

Submitted by:

Mustafa Hasan (TL),

Rajat Kaliya, Kartik Sharma,
Kapil Dev Sharma,
Ashu Hasan

Submitted to:

Shouhaddo Paul (mentor)

Mr. Shubham Tandon,
Managing Director, ConsoleLancer

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. Recognized by 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

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 .

<u>Transformation of images</u>

- 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.

Skimage

What's the first thing that comes to your mind when you hear "image preprocessing"? I received a few quizzical looks when I asked this question to a group of data science enthusiasts. If you're wondering what this is, read on! We're pretty familiar with the preprocessing steps for structured (tabular) data. You find and plug in any missing values, detect and deal with outliers, etc. This helps us build better and more robust machine learning models. But how does that work when we're working with image data?

As it turns out, the preprocessing step is a crucial one in the world of computer vision (images, videos, and so on). skimage, part of the scikit-learn family, is a really helpful library to get us started.

Scikit-image, or skimage, is an open source Python package designed for image preprocessing.

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.

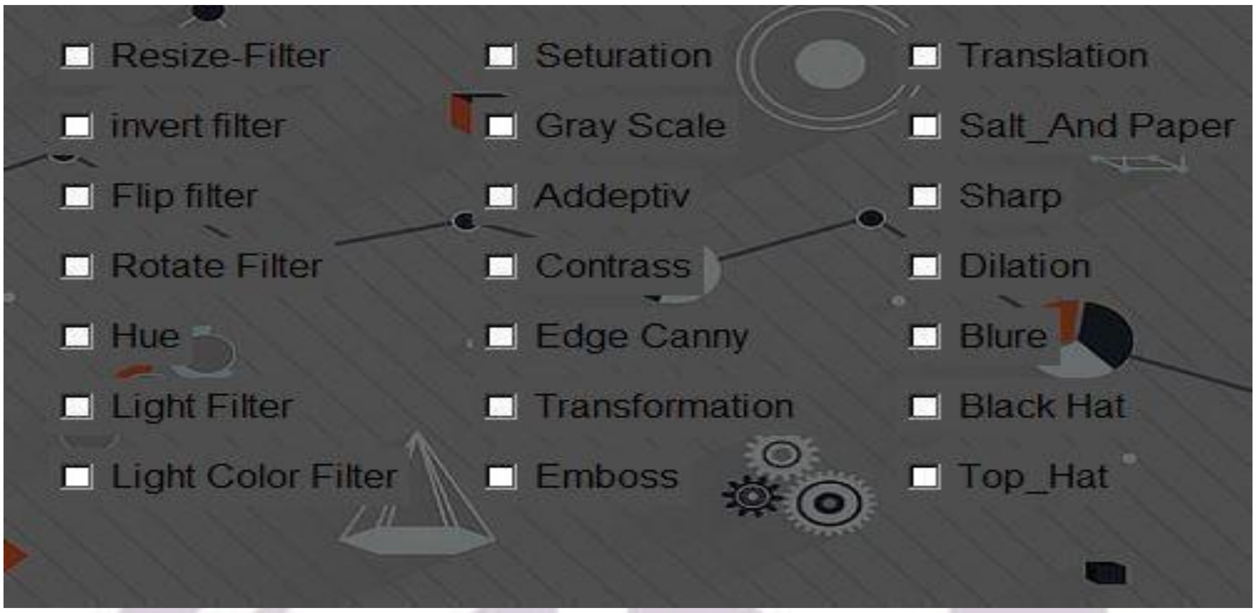
Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

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
- Addeptive 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



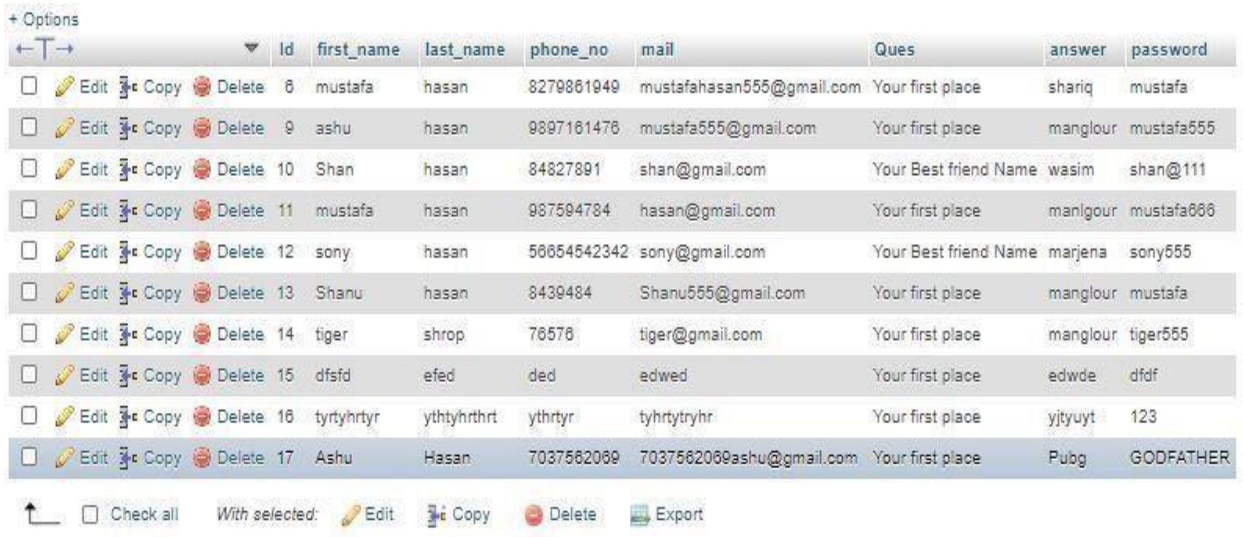
My sql

MySQL is one of the most popular database management systems (DBMSs) on the market today. It ranked second only to the Oracle DBMS in this year's DBEngines Ranking. As 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 loops, functions, exception handling, and installing Python packages using pip. You should also have a basic understanding of relational database management systems and SQL queries like SELECT, DROP, CREATE, and JOIN.

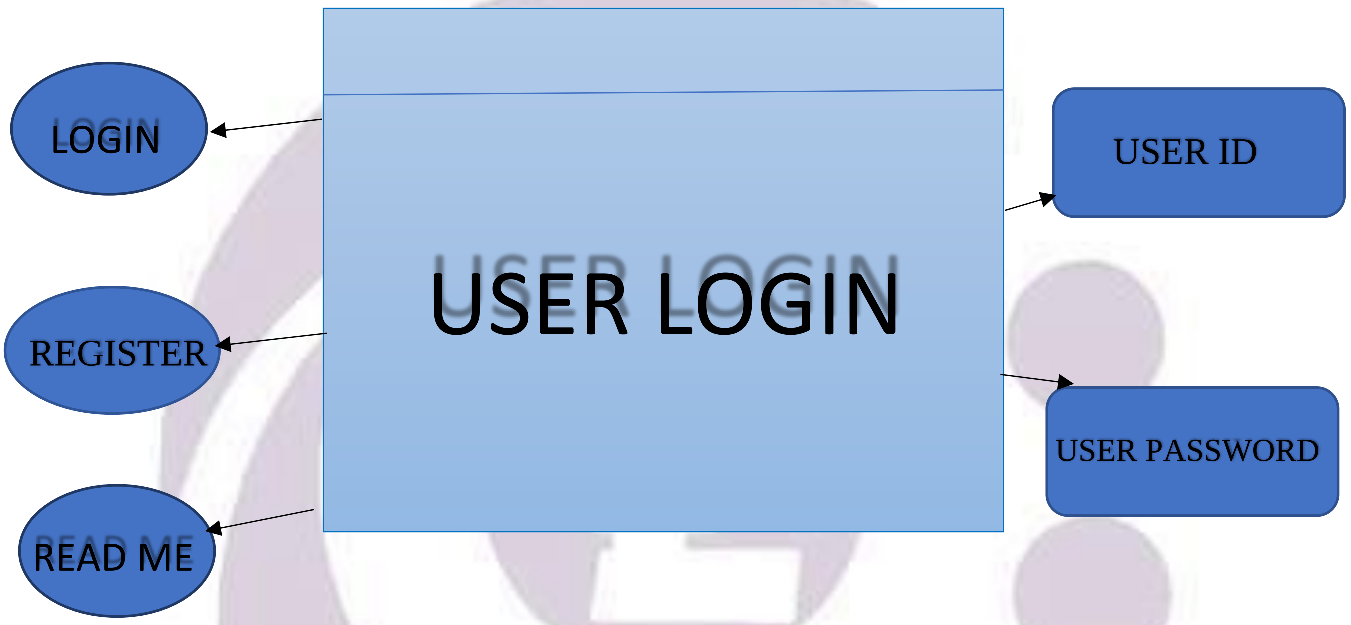


+ Options

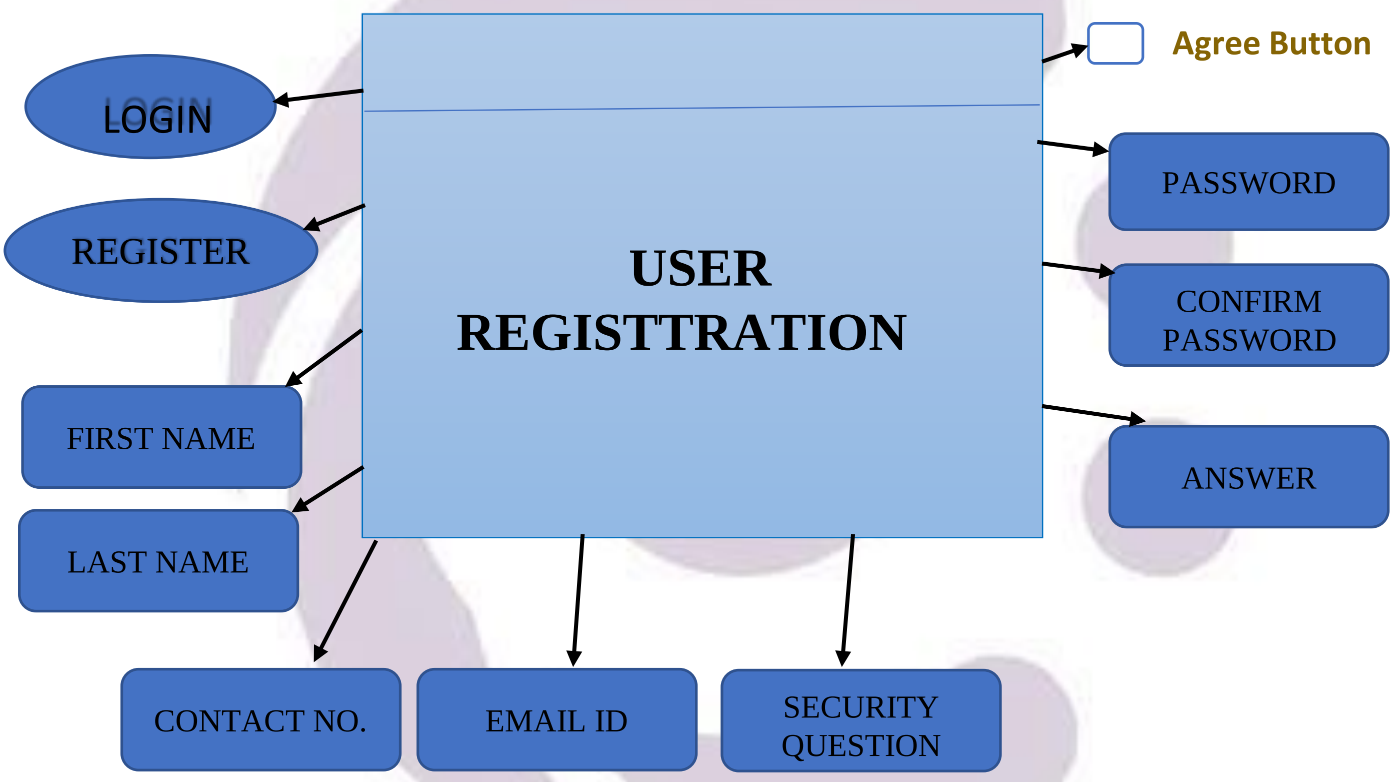
←T→				Id	first_name	last_name	phone_no	mail	Ques	answer	password
☐	Edit	Copy	Delete	6	mustafa	hasan	8279861949	mustafahasan555@gmail.com	Your first place	shariq	mustafa
☐	Edit	Copy	Delete	9	ashu	hasan	9897161476	mustafa555@gmail.com	Your first place	manglour	mustafa555
☐	Edit	Copy	Delete	10	Shan	hasan	84827891	shan@gmail.com	Your Best friend Name	wasim	shan@111
☐	Edit	Copy	Delete	11	mustafa	hasan	987594784	hasan@gmail.com	Your first place	manlgour	mustafa666
☐	Edit	Copy	Delete	12	sony	hasan	56654542342	sony@gmail.com	Your Best friend Name	marjena	sony555
☐	Edit	Copy	Delete	13	Shanu	hasan	8439484	Shanu555@gmail.com	Your first place	manglour	mustafa
☐	Edit	Copy	Delete	14	tiger	shrop	76576	tiger@gmail.com	Your first place	manglour	tiger555
☐	Edit	Copy	Delete	15	dfsfd	efed	ded	edwed	Your first place	edwde	dfdf
☐	Edit	Copy	Delete	16	tyrtyhrtyr	ythtyhrthrt	ythrtyr	tyhrtytryhr	Your first place	yjtyuyt	123
☐	Edit	Copy	Delete	17	Ashu	Hasan	7037562069	7037562069ashu@gmail.com	Your first place	Pubg	GODFATHER

USER PHASE

1.WINDOW

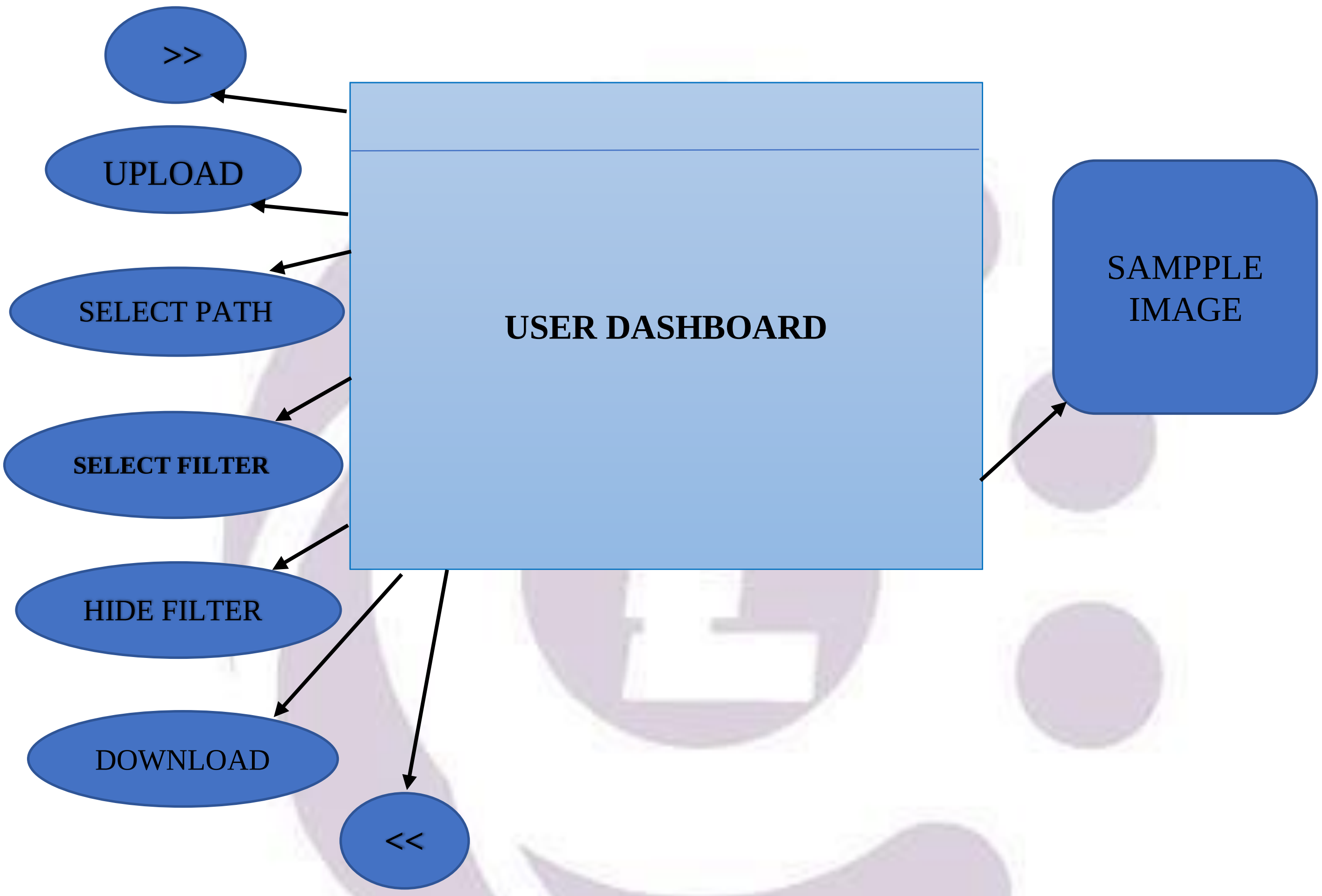


2.WINDOW

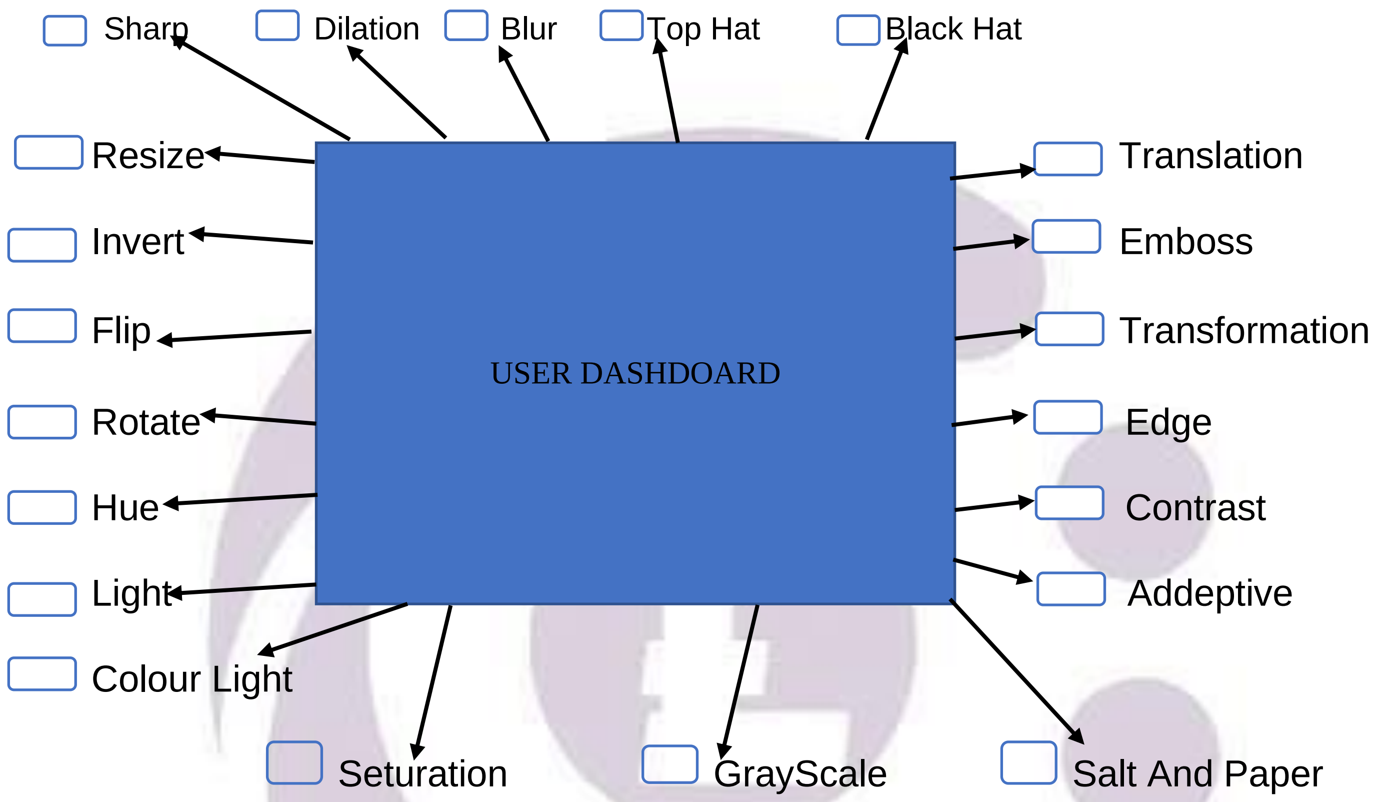


USER REGISTTRATION
Agree Button
LOGIN
REGISTER
FIRST NAME
LAST NAME
CONTACT NO.
EMAIL ID
SECURITY QUESTION
PASSWORD
CONFIRM PASSWORD
ANSWER

3.WINDOW

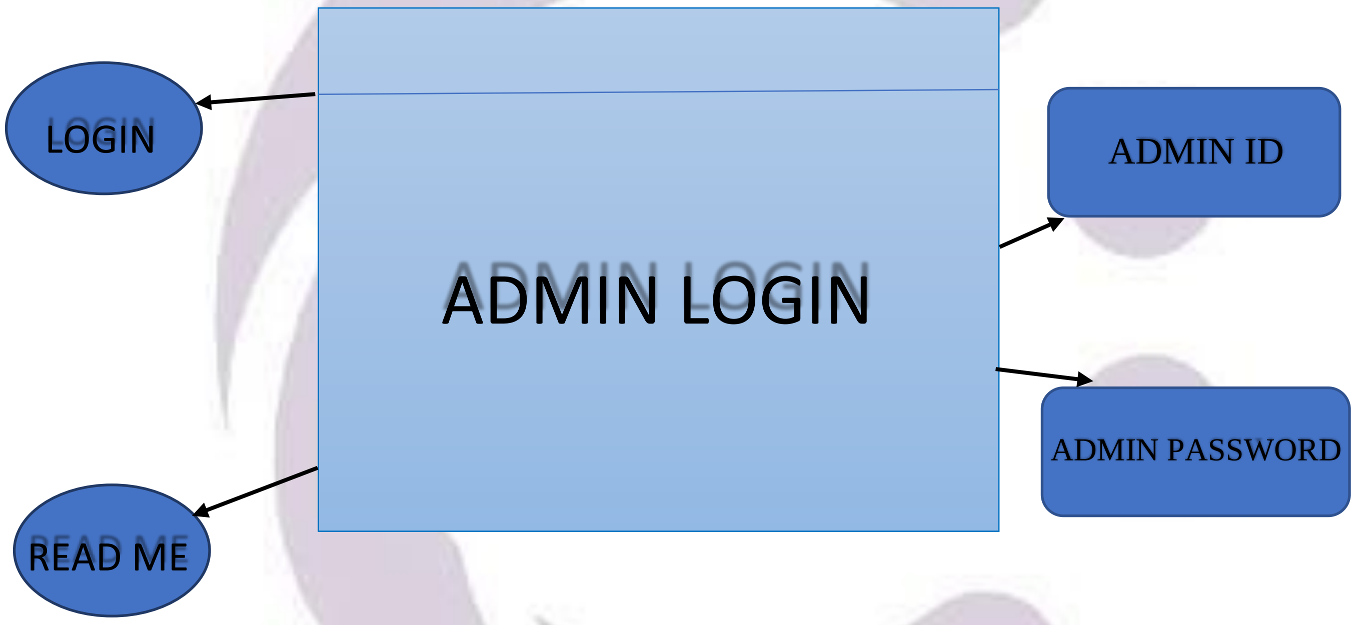

>>
UPLOAD
SELECT PATH
SELECT FILTER
HIDE FILTER
DOWNLOAD
<<
USER DASHBOARD
SAMPPLE IMAGE

4.WINDOW

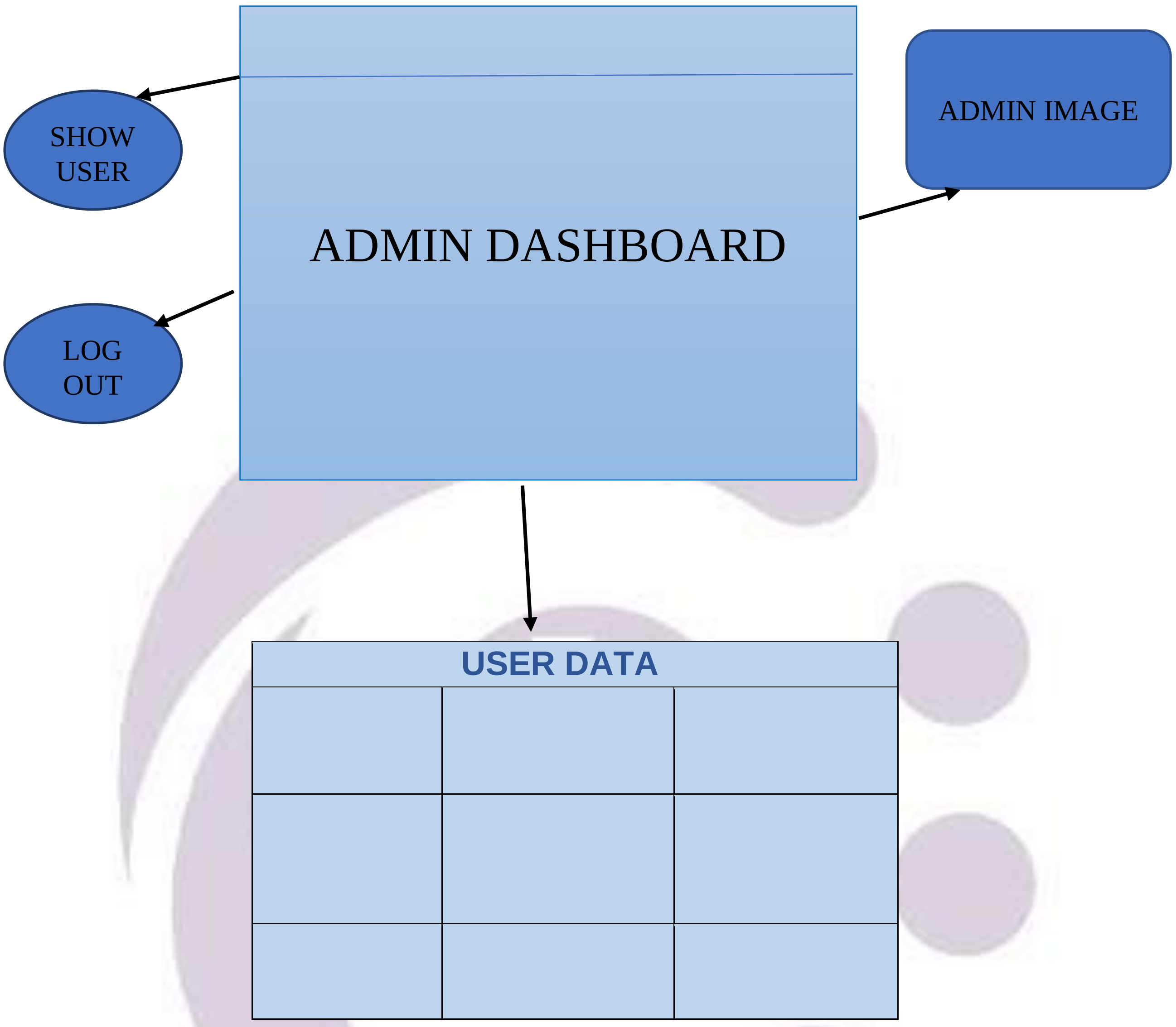


Sharp
Dilation
Blur
Top Hat
Black Hat
Resize
Invert
Flip
Rotate
Hue
Light
Colour Light
USER DASHDOARD
Translation
Emboss
Transformation
Edge
Contrast
Addeptive
Seturation
GrayScale
Salt And Paper

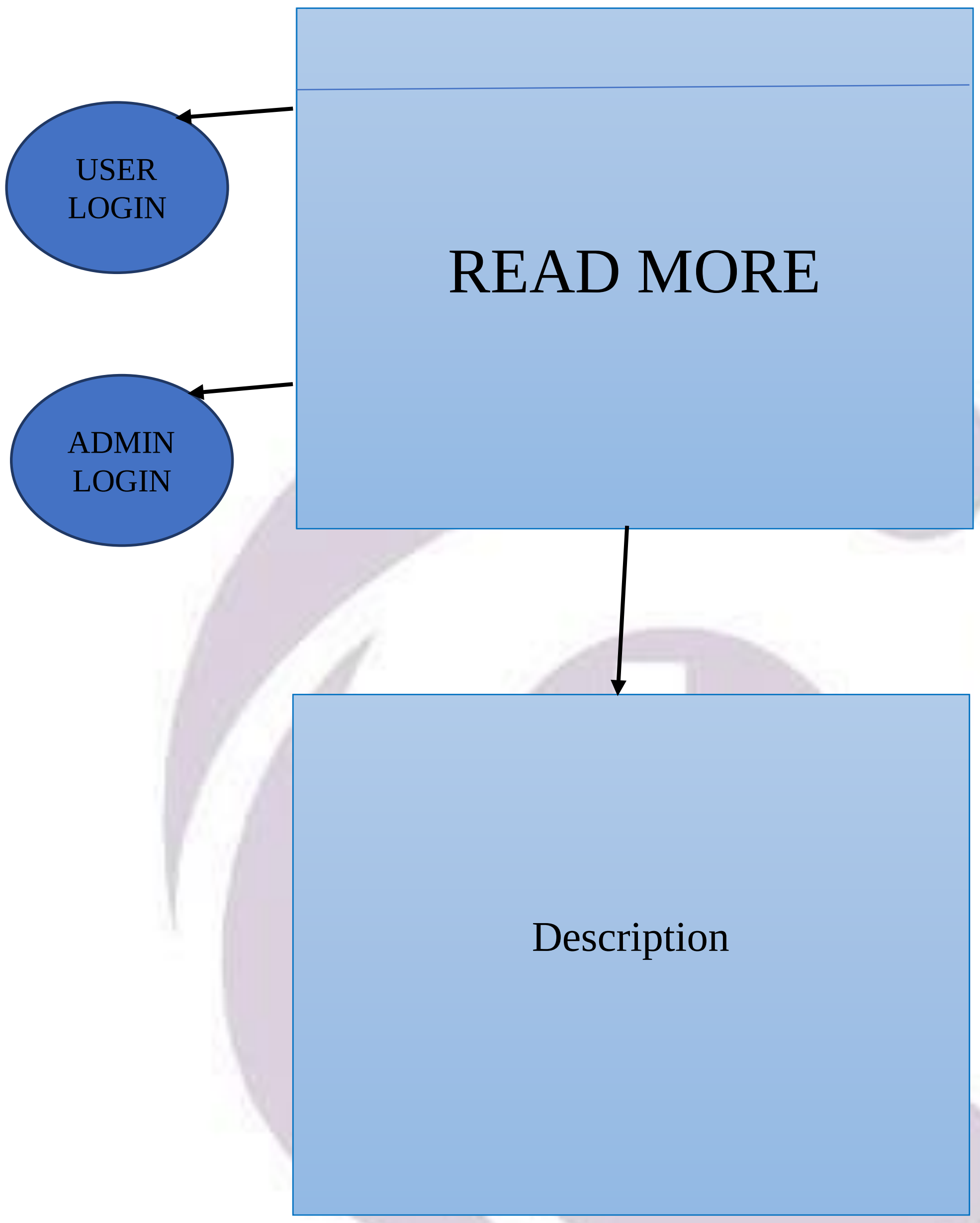
5.WINDOW



6.WINDOW



7.WINDOW



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)
```

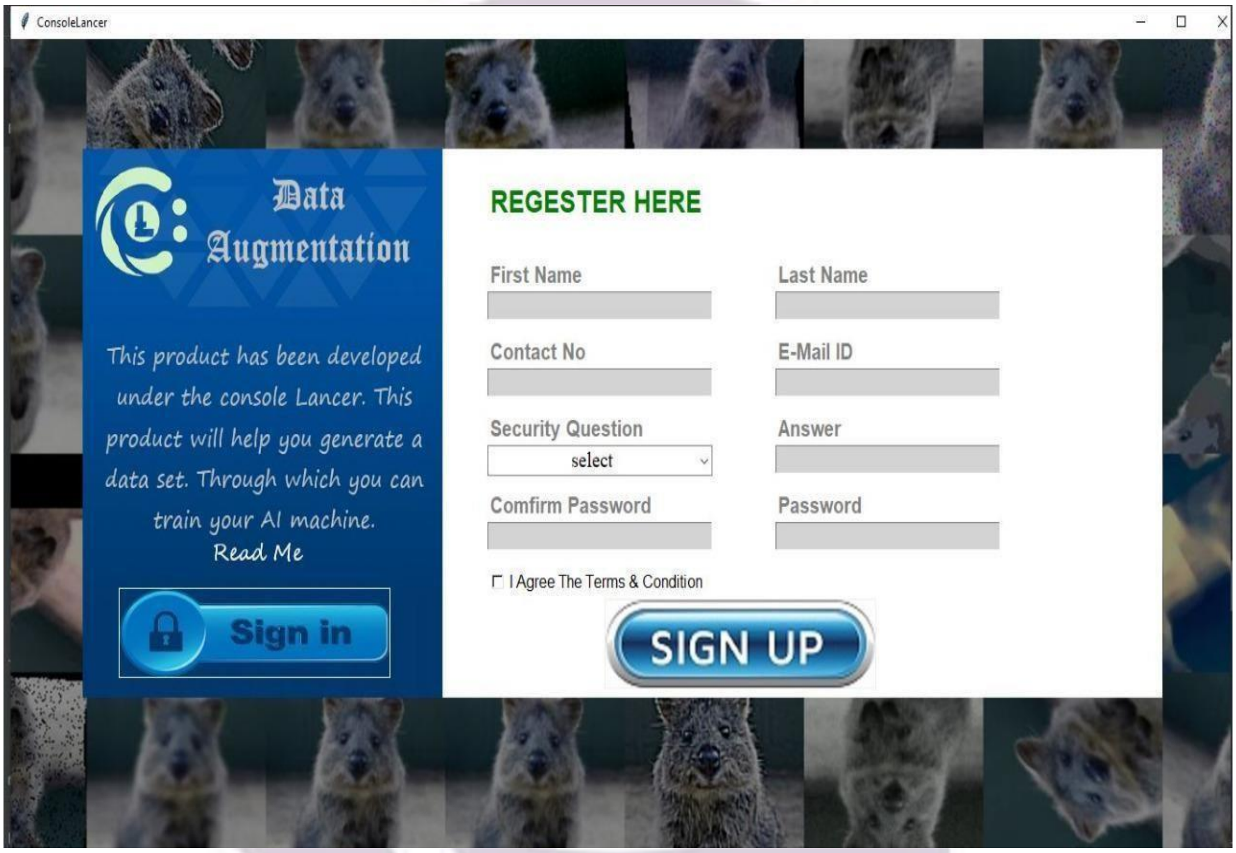
```
        # _____Download:- Creating button to generate data
set==============================
        self.generate = ImageTk.PhotoImage(file="Download.png")
        Generate_Sample = Button(self.root, image=self.generate, bg="#4e4e4e",
activebackground="#4e4e4e",borderwidth=0, cursor="hand2", command=download_Button)


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()
```

SCREENSHOTS

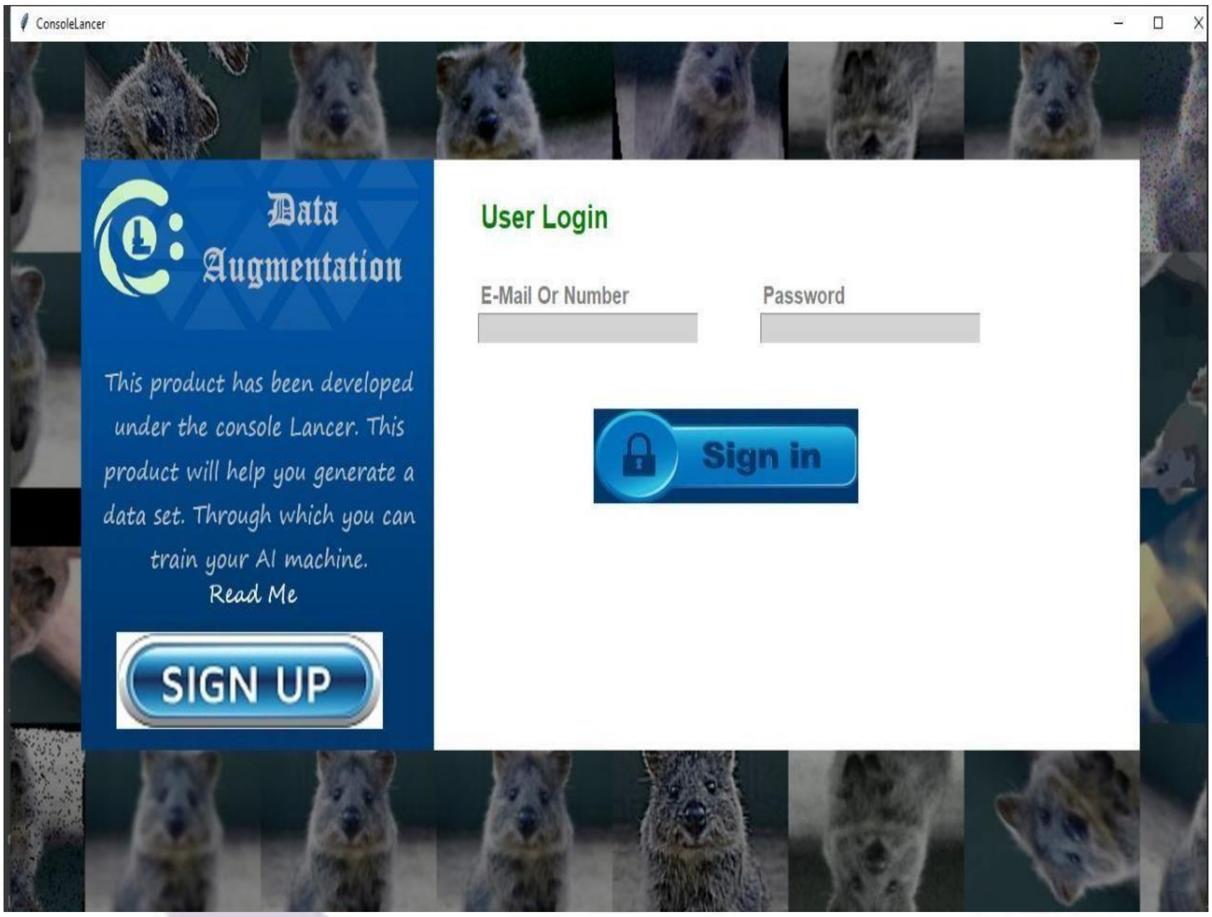


This product has been developed under the console Lancer. This product will help you generate a data set. Through which you can train your AI machine.
Read Me

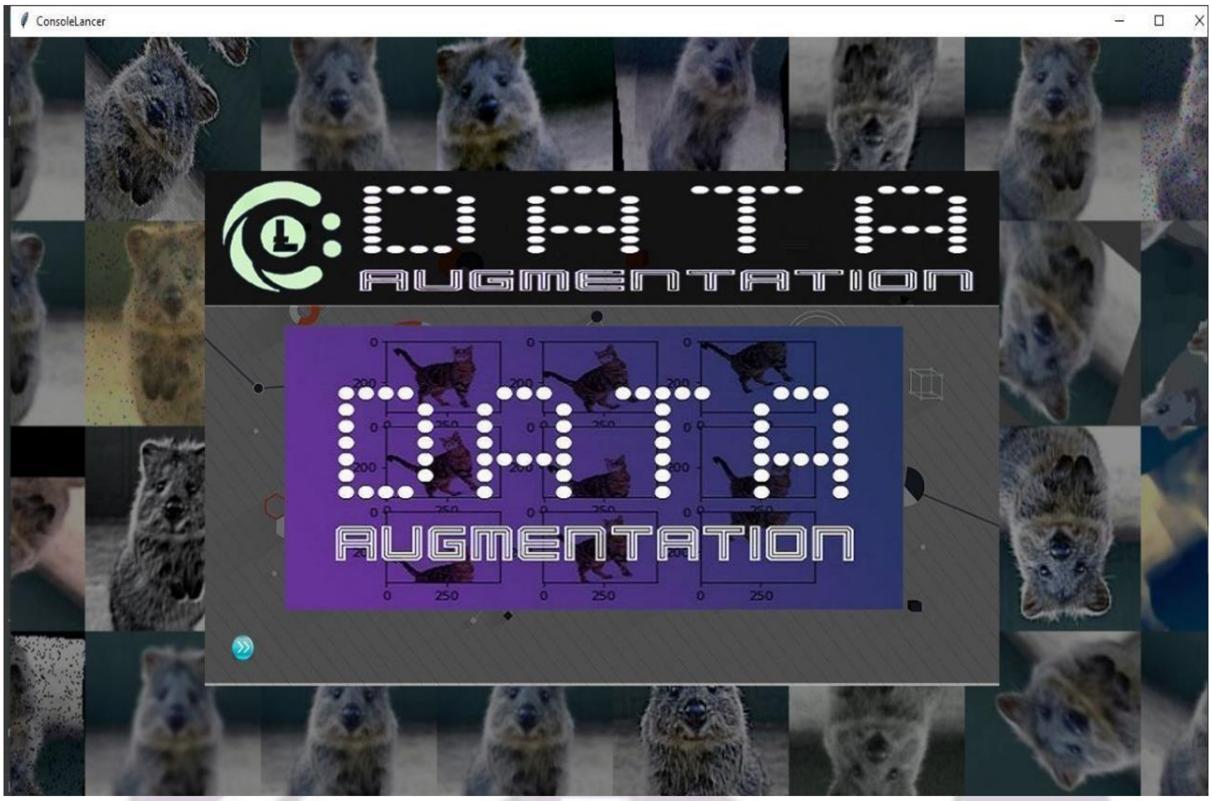
REGESTER HERE

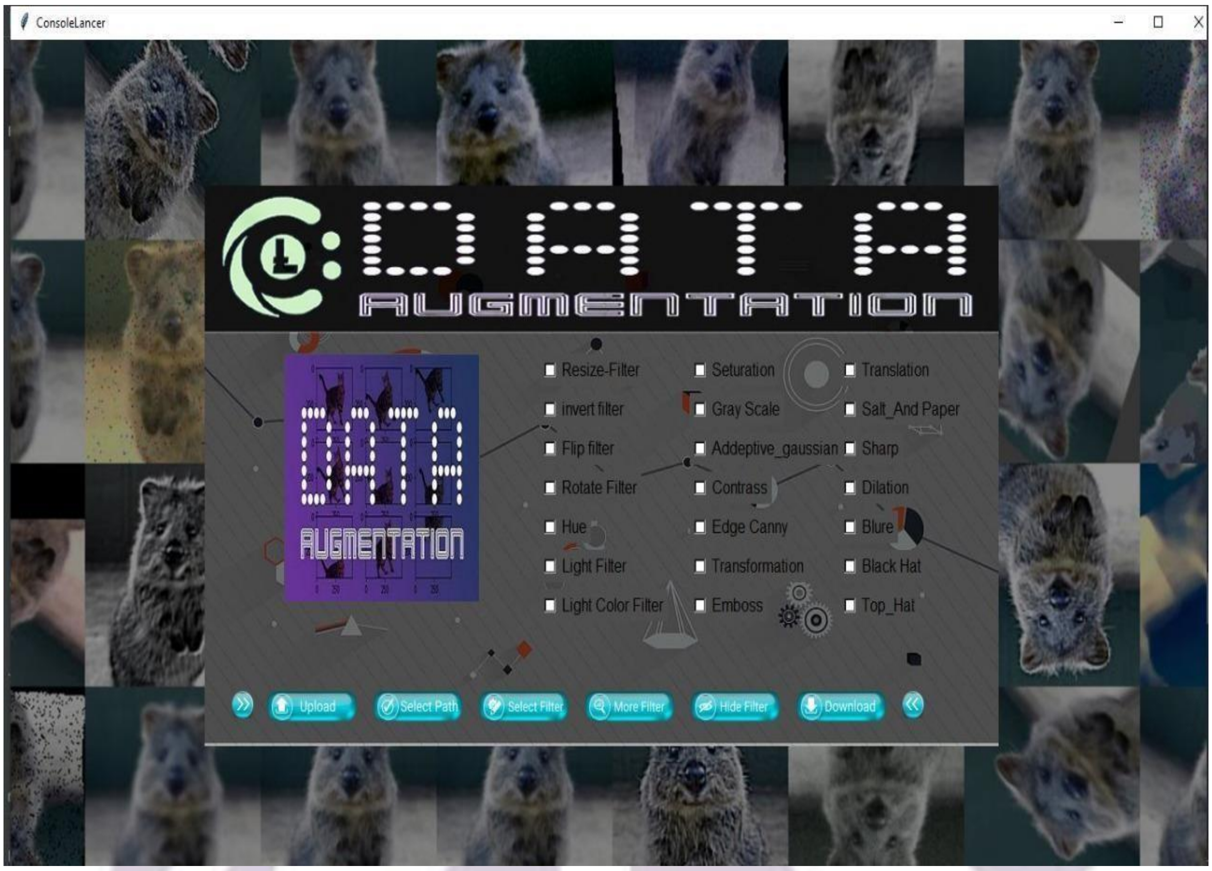
First Name	Last Name
Contact No	E-Mail ID
Security Question: select	Answer
Comfirm Password	Password

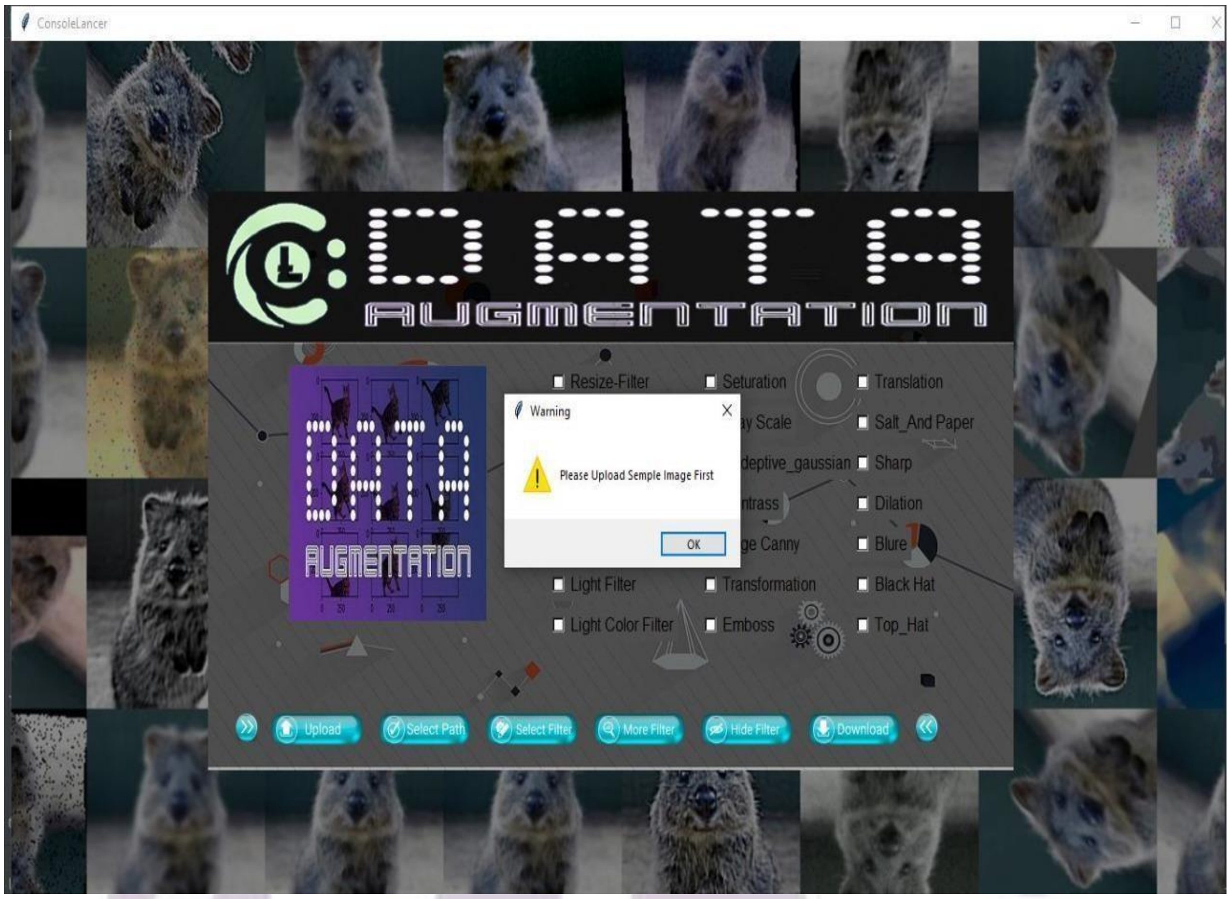
☐ I Agree The Terms & Condition



This product has been developed under the console Lancer. This product will help you generate a data set. Through which you can train your AI machine.


ConsoleLancer
DATA
AUGMENTATION
DATA
AUGMENTATION

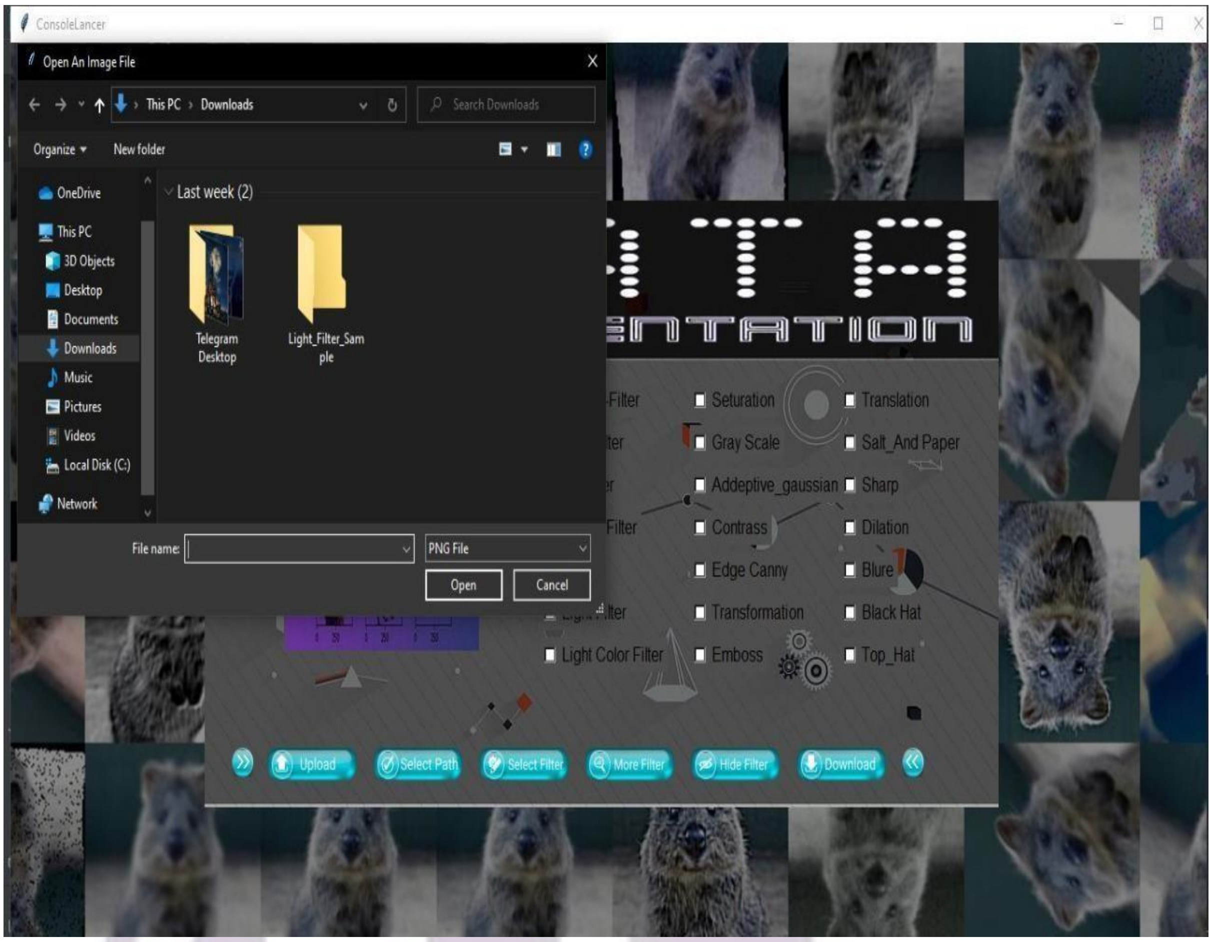

ConsoleLancer
DATA AUGMENTATION
Resize-Filter
Seturation
Translation
invert filter
Gray Scale
Salt_And Paper
Flip filter
Addeptive_gaussian
Sharp
Rotate Filter
Contrass
Dilation
Hue
Edge Canny
Blure
Light Filter
Transformation
Black Hat
Light Color Filter
Emboss
Top_Hat
Upload
Select Path
Select Filter
More Filter
Hide Filter
Download

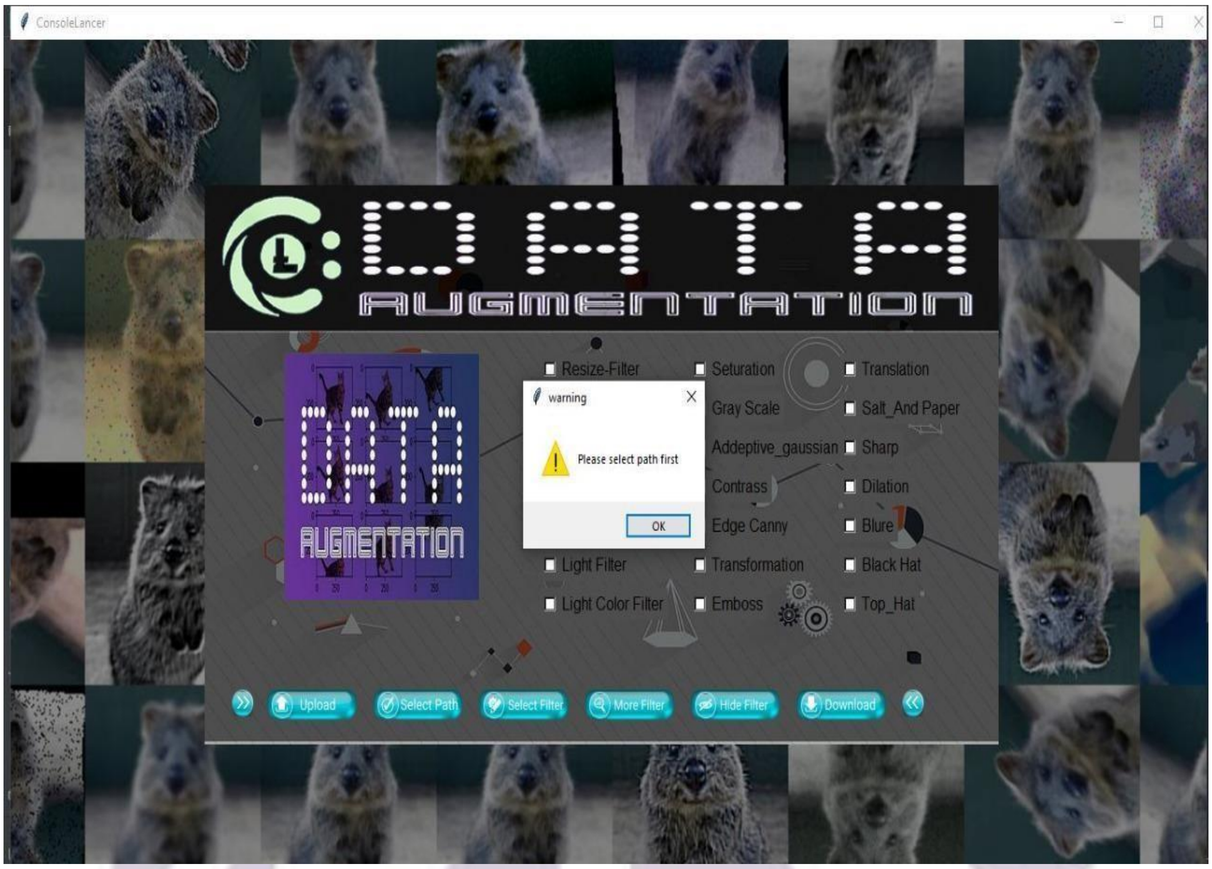


Warning

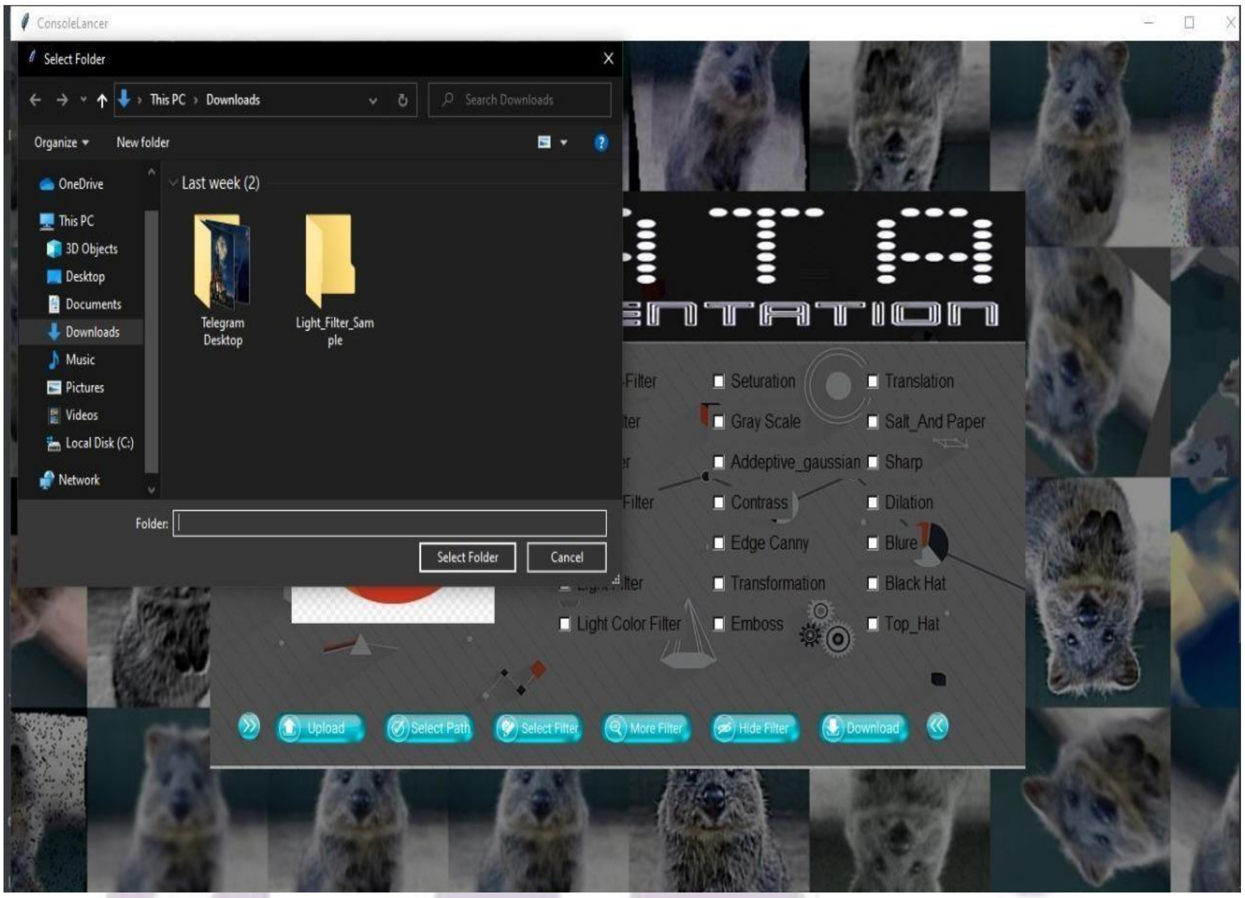
Please Upload Semple Image First

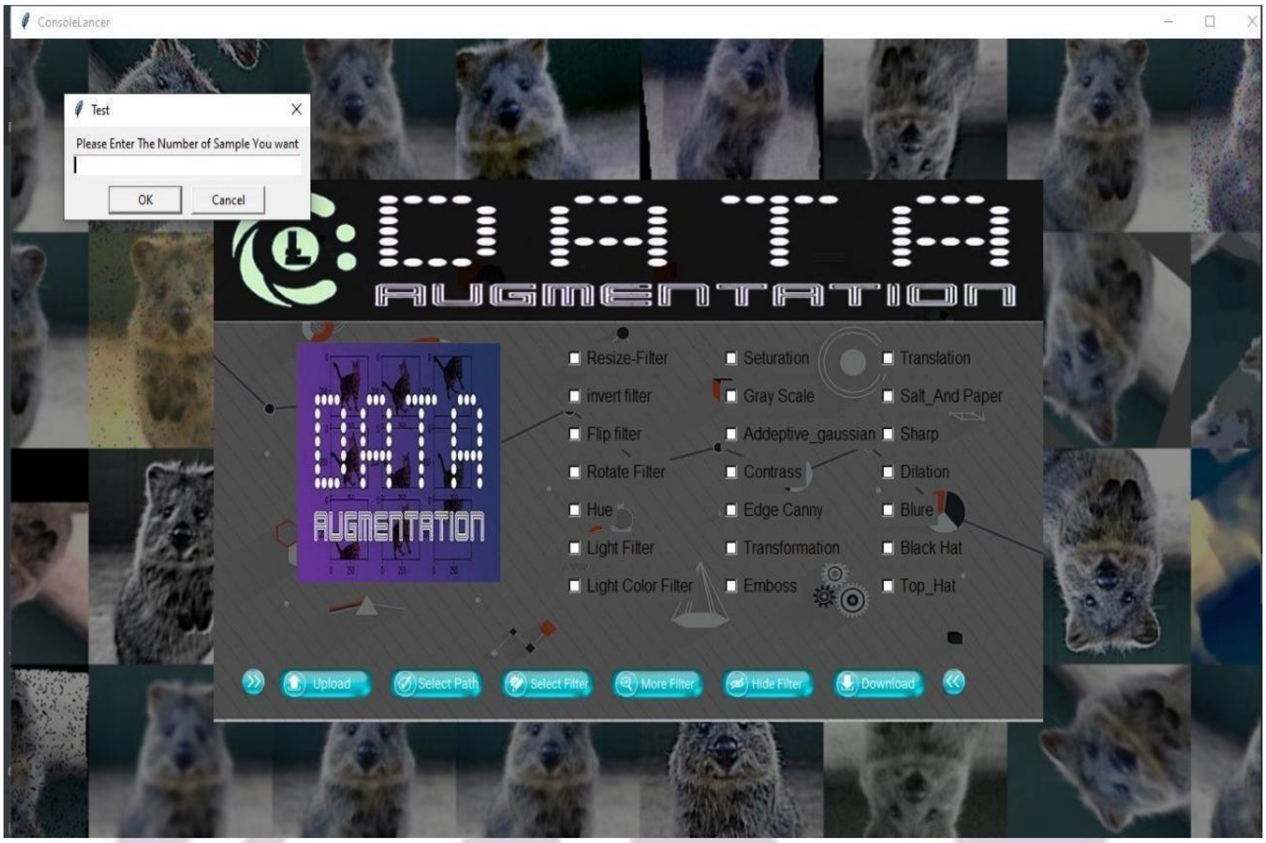
OK

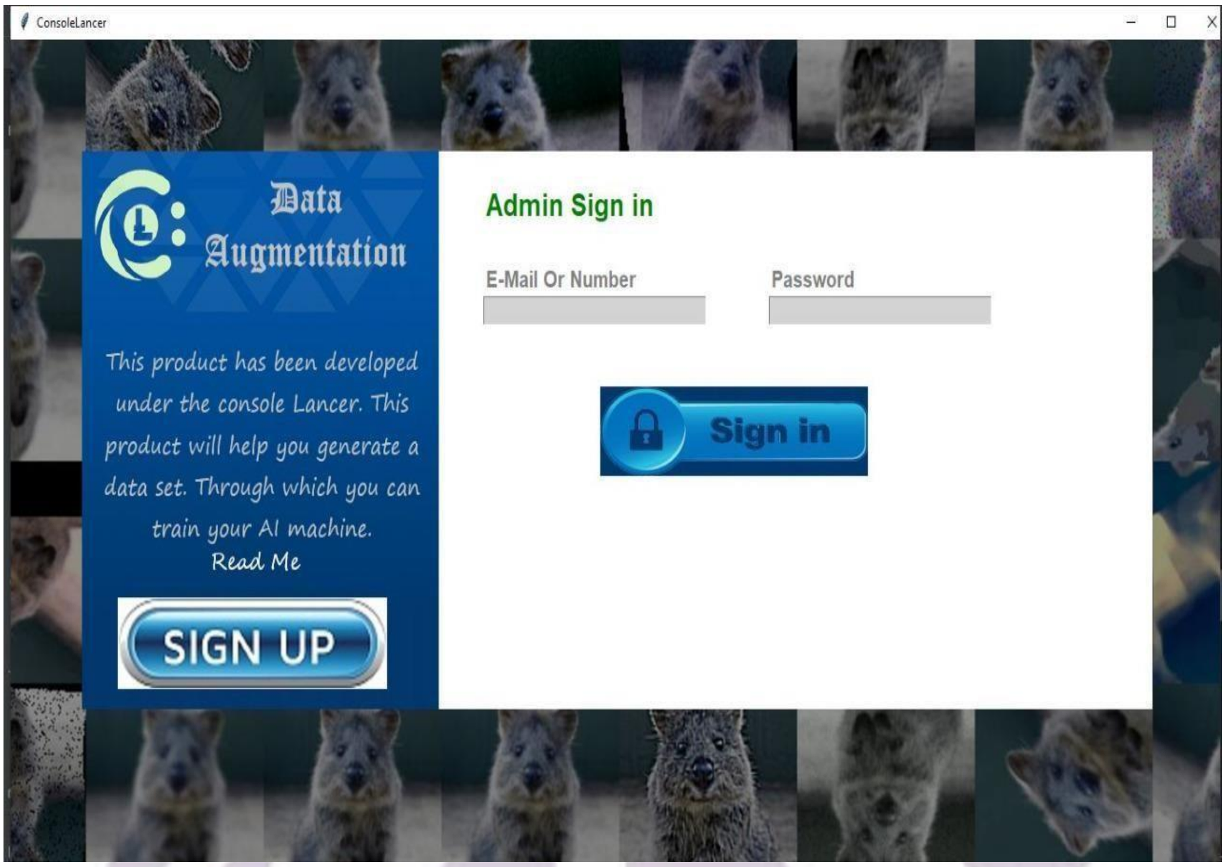

ConsoleLancer
Open An Image File
This PC › Downloads
Search Downloads
Organize
New folder
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Network
Last week (2)
Telegram Desktop
Light_Filter_Sample
File name:
PNG File
Open
Cancel
Seturation
Translation
Gray Scale
Salt_And Paper
Addeptive_gaussian
Sharp
Contrass
Dilation
Edge Canny
Blure
Transformation
Black Hat
Light Color Filter
Emboss
Top_Hat
Upload
Select Path
Select Filter
More Filter
Hide Filter
Download



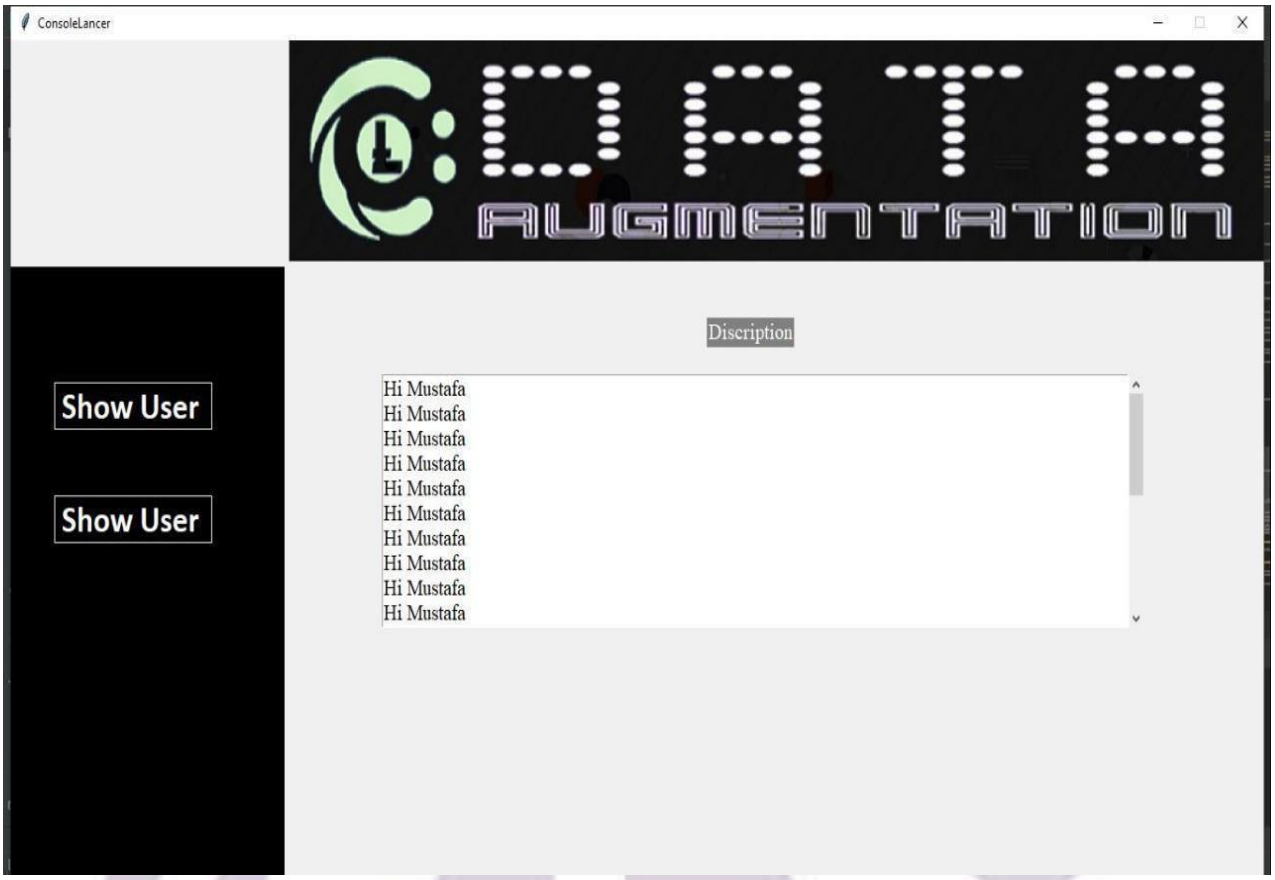
Please select path first


ConsoleLancer
Select Folder
This PC › Downloads
Search Downloads
Organize
New folder
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Network
Last week (2)
Telegram Desktop
Light_Filter_Sample
Folder:
Select Folder
Cancel
Seturation
Translation
Gray Scale
Salt_And Paper
Addeptive_gaussian
Sharp
Contrass
Dilation
Edge Canny
Blure
Transformation
Black Hat
Light Color Filter
Emboss
Top_Hat
Upload
Select Path
Select Filter
More Filter
Hide Filter
Download

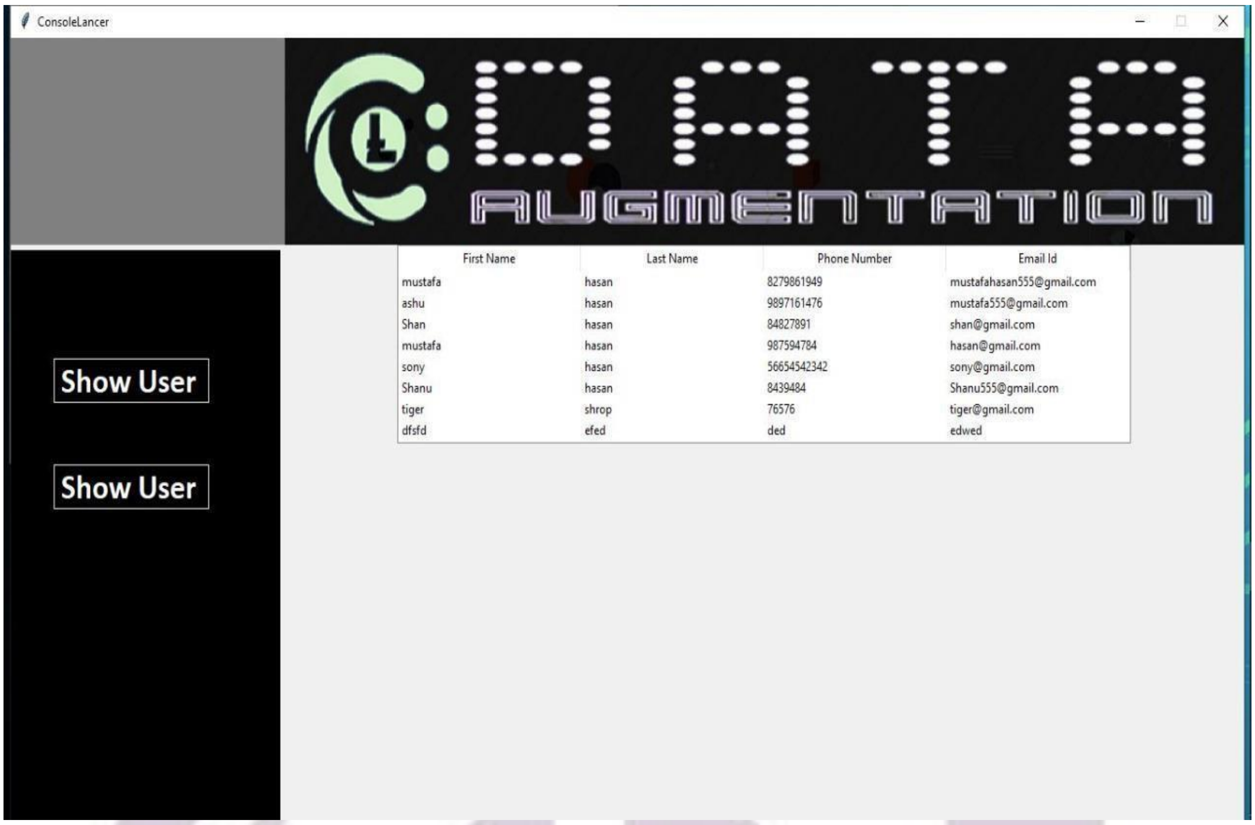

ConsoleLancer
Test
Please Enter The Number of Sample You want
OK
Cancel
DATA
AUGMENTATION
Resize-Filter
invert filter
Flip filter
Rotate Filter
Hue
Light Filter
Light Color Filter
Seturation
Gray Scale
Addeptive_gaussian
Contrass
Edge Canny
Transformation
Emboss
Translation
Salt_And Paper
Sharp
Dilation
Blure
Black Hat
Top_Hat
Upload
Select Path
Select Filter
More Filter
Hide Filter
Download



This product has been developed under the console Lancer. This product will help you generate a data set. Through which you can train your AI machine.



```
Hi Mustafa
Hi Mustafa
Hi Mustafa
Hi Mustafa
Hi Mustafa
Hi Mustafa
Hi Mustafa
Hi Mustafa
Hi Mustafa
Hi Mustafa
```



First Name	Last Name	Phone Number	Email Id
mustafa	hasan	8279861949	mustafahasan555@gmail.com
ashu	hasan	9897161476	mustafa555@gmail.com
Shan	hasan	84827891	shan@gmail.com
mustafa	hasan	987594784	hasan@gmail.com
sony	hasan	56654542342	sony@gmail.com
Shanu	hasan	8439484	Shanu555@gmail.com
tiger	shrop	76576	tiger@gmail.com
dfsfd	efed	ded	edwed