Drowsiness detection and Encryption of Results

Presented by

Gottumukkala Sai (122004080)
Chukkaluru Mahidhar Reddy (122004064)

Project Guide
Dr. Ganapathy R - (AP III)
School of Electrical and Electronics Engineering
SASTRA Deemed to be University

Motivation

- It is difficult for any teacher to monitor all the students during the online class session.
- This reason has triggered us to work on a solution that can continuously monitor and detect drowsiness on their faces if any.
- This requirement can be achieved using a Deep Learning.

Objective

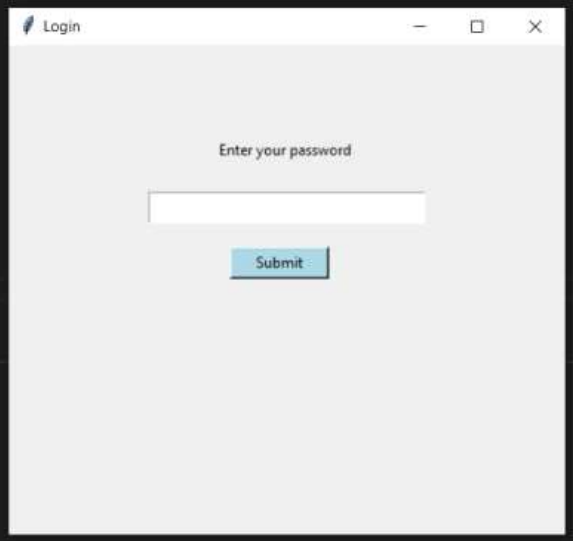
- To detect drowsiness on the faces of students during online class sessions using Deep Learning and encrypt the photos of students in drowsy state.

Graphical User Interface

- A graphical user interface (GUI) is a type of user interface through which users interact with electronic devices via visual indicator representations.
- For taking user input and processing we created login window.
- In this project we used **Tkinter** for implementing **GUI**.
- Steps to create a tkinter window:
    1. Importing the module – tkinter
    2. Create the main window (container)
    3. Add any number of widgets to the main window
    4. Apply the event Trigger on the widgets.

GUI Window

```python
from tkinter import *
from tkinter import messagebox
import os
import os.path
from os.path import isfile
import time

def checkpwd():
    pwd = e.get()
    if(len(pwd)==0):
        f=0
        messagebox.showinfo("Warning", "Empty password is not accepted")
    print(pwd)
    f = open("D:\Drowsiness detection\ptext.txt", "r")
    p = f.read()
    f.close()
    if(p == pwd):
        frame.destroy()
        exec(open("D:\Drowsiness detection\drowsinessdetection.py").read())
    else:
        if f==1:
```



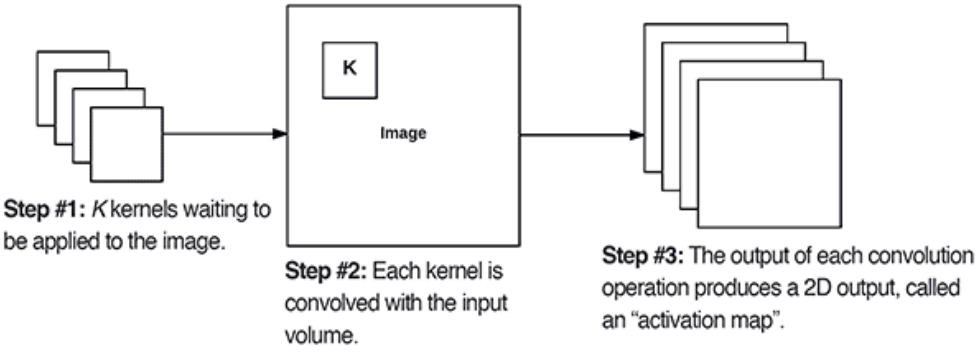

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\saine> python -u "d:\Drowsiness detection\GUI.py"
```

MODEL

- In this project we used sequential model for implementing drowsiness detection.
- Sequence models are the machine learning models that input or output sequences of data. Sequential data includes text streams, audio clips, video clips, time-series data.
- It allows you to build a model layer by layer. We use the 'add()' function to add layers to our model. Our first 2 layers are Conv2D layers. These are convolution layers that will deal with our input images, which are seen as 2-dimensional matrices.
- For implementing these models we imported Keras.

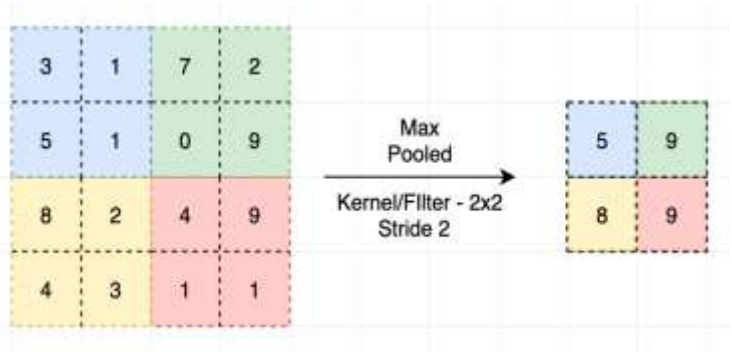
CONVOLUTIONAL 2D LAYER

- Keras Conv2D is a 2D Convolution Layer, this layer creates a convolution kernel that is wind with layers input which helps produce a tensor of outputs.
- In image processing, kernel is a convolution matrix or masks which can be used for blurring, sharpening, embossing, edge detection, and more by doing a convolution between a kernel and an image.
- This layer creates a convolution kernel that is wind with layers input which helps produce a tensor of outputs.



MAX POOLING

- A pooling layer is another building block of a CNN. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network.
- Pooling layer operates on each feature map independently.
- Max Pooling calculates the maximum value for patches of a feature map, and uses it to create a downsampled (pooled) feature map. It is usually used after a convolutional layer.
- Global max pooling = ordinary max pooling layer with pool size equals to the size of the input (minus filter size + 1, to be precise).

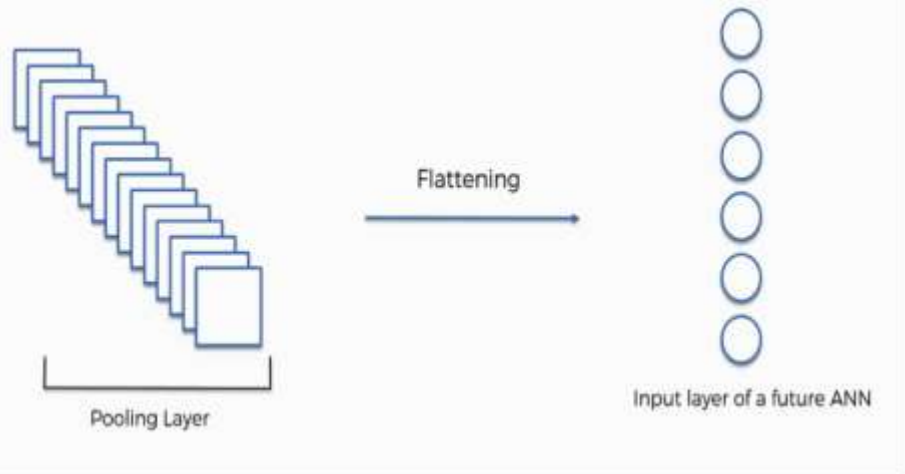


➢ DROPOUT:

1. Dropout is a technique where randomly selected neurons are ignored during training. They are “dropped-out” randomly. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass.
2. A Simple Way to Prevent Neural Networks from Overfitting.
3. The outputs of a layer under dropout are randomly subsampled, it has the effect of reducing the capacity or thinning the network during training. As such, a wider network, e.g. more nodes, may be required when using dropout.

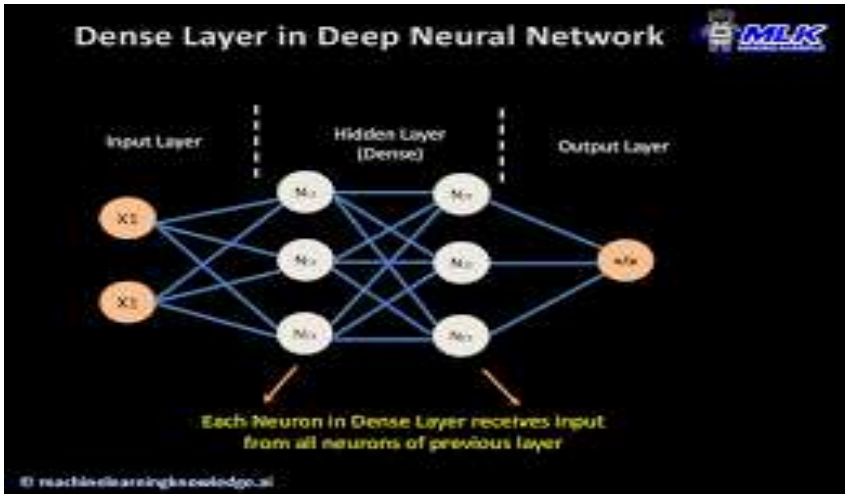
➢ FLATTEN:

1. Flattening is converting the data into a 1-dimensional array for inputting it to the next layer. We flatten the output of the convolutional layers to create a single long feature vector. And it is connected to the final classification model, which is called a fully-connected layer.



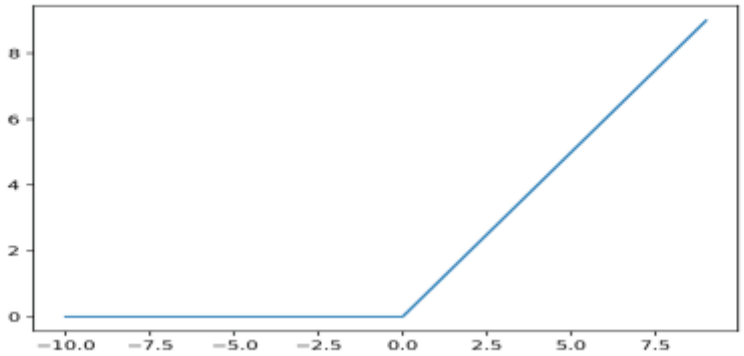
➢ DENSE:

1. The dense layer is a neural network layer that is connected deeply, which means each neuron in the dense layer receives input from all neurons of its previous layer. The dense layer is found to be the most commonly used layer in the models.
2. Dense Layer is used to classify image based on output from convolutional layers.



➢ ACTIVATION-RELU:

1. RELU stands for Rectified Linear Unit Activation.
2. Activation function is a function that is added into an artificial neural network in order to help the network learn complex patterns in the data.
3. The rectified linear activation function or ReLU for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero.



Drowsiness Detection

Steps to be followed:

a. Take image as input from a camera.

b. Detect the face in the image and create a Region of Interest (ROI).

c. Detect the eyes from ROI and feed it to the classifier.

d. Classifier will categorize whether eyes are open or closed.

e. Calculate score to check whether the person is drowsy.

f. If the score exceeds the threshold, we capture the face.

- **Algorithm used: CNN**

- **What is CNN?**

  Convolutional Neural Networks(CNN) is a special type of deep neural network which performs extremely well for image classification purposes. A CNN basically consists of an input layer,an output layer and a hidden layer which can have multiple layers.
  A convolutional operation is performed on these layers using a filter that performs 2D matrix multiplication on the layer and filter.

The CNN model architecture consists of the following layers:

- Convolution layer;32 nodes,kernel size 3
- Convolution layer;32 nodes,kernel size 3
- Convolution layer;64 nodes,kernal size 3
- Fully connected layer;128 nodes

The final layer is also a fully connected layer with 2 nodes. A Relu activation function is used in all the layer except the output layer in which we use softmax

Advantages of CNN:

- Very High accuracy in image recognition problems.
- Automatically detects the important features without any human supervision.
- Weight sharing.

Project Requirements:

The requirement for this Python project is a webcam through which we will capture images. You need to have Python (3.6 version recommended) installed on your system, then using pip, you can install the necessary packages.

Required packages:

- OpenCV – pip install opencv-python (face and eye detection).
- TensorFlow – pip install tensorflow (keras uses TensorFlow as backend).
- Keras – pip install keras (to build our classification model).

Detailed step explantion:

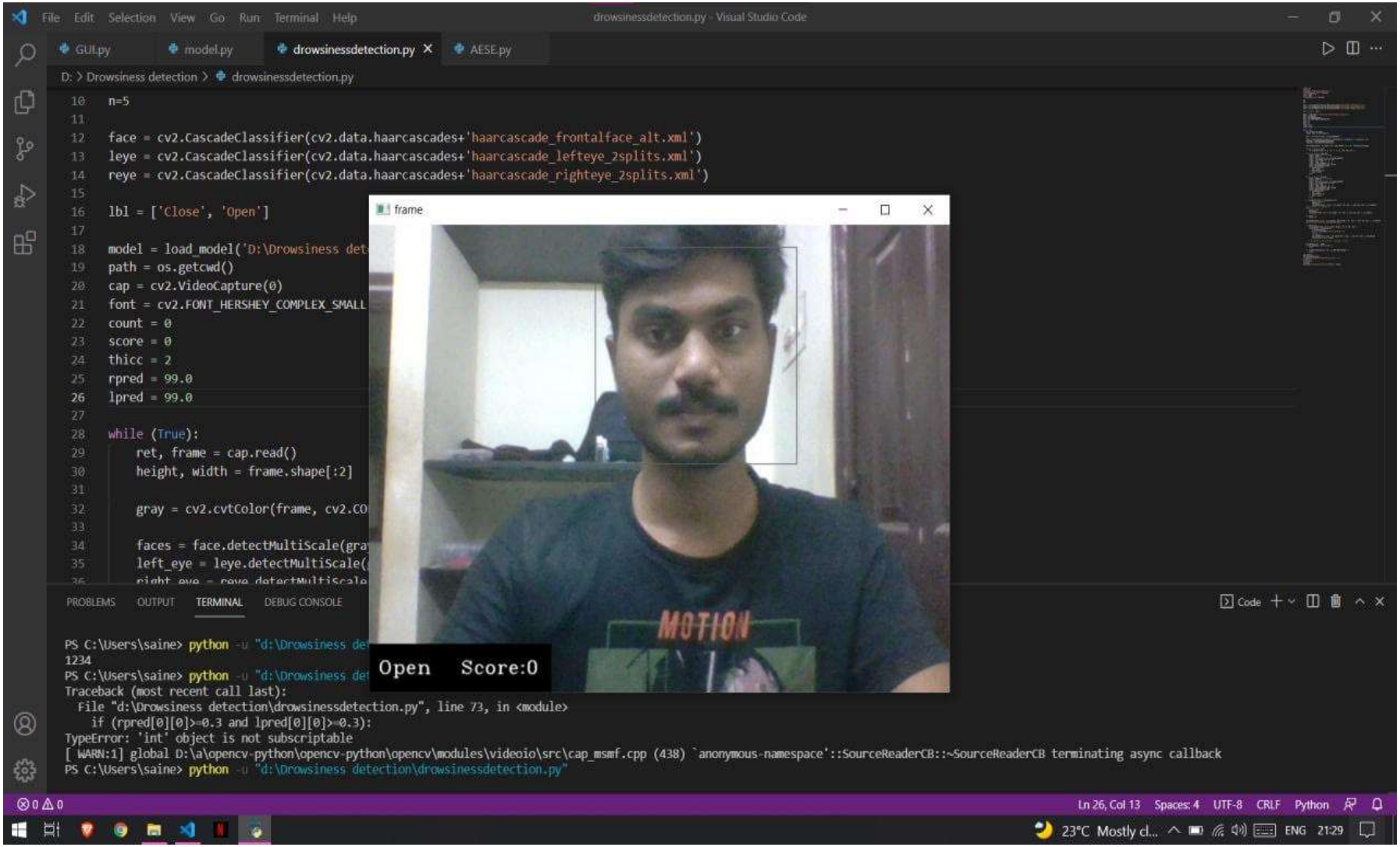
- **Step 1** – Take Image as Input from a CameraWith a webcam, we will take images as input. So to access the webcam, we made an infinite loop that will capture each frame. We use the method provided by OpenCV, cv2.VideoCapture(0) to access the camera and set the capture object (cap). cap.read() will read each frame and we store the image in a frame variable.
-

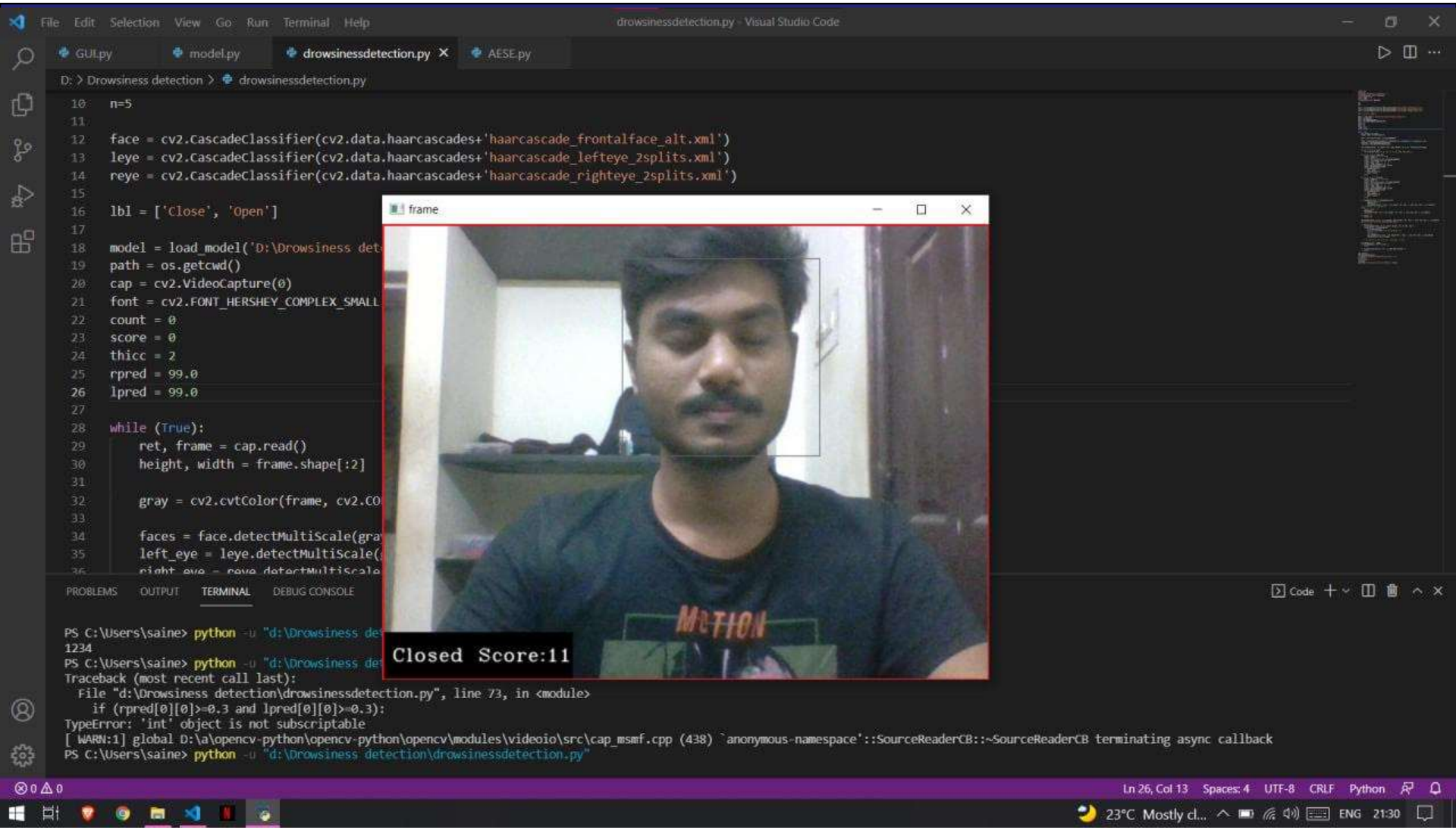
- **Step 2 – Detect Face in the Image and Create a Region of Interest (ROI)**
- To detect the face in the image, we need to first convert the image into grayscale as the OpenCV algorithm for object detection takes gray images in the input. We don't need color information to detect the objects. We will be using haar cascade classifier to detect faces. This line is used to set our classifier face = cv2.CascadeClassifier(' path to our haar cascade xml file'). Then we perform the detection using faces = face.detectMultiScale(gray). It returns an array of detections with x,y coordinates, and height, the width of the boundary box of the object. Now we can iterate over the faces and draw boundary boxes for each face.

- **Step 3 – Detect the eyes from ROI and feed it to the classifier**
- The same procedure to detect faces is used to detect eyes. First, we set the cascade classifier for eyes in leye and reye respectively then detect the eyes using left_eye = leye.detectMultiScale(gray). Now we need to extract only the eyes data from the full image. This can be achieved by extracting the boundary box of the eye and then we can pull out the eye image from the frame with this code.

- **Step 4 – Classifier will Categorize whether Eyes are Open or Closed**
- We are using CNN classifier for predicting the eye status. To feed our image into the model, we need to perform certain operations because the model needs the correct dimensions to start with. First, we convert the color image into grayscale using **r_eye = cv2.cvtColor(r_eye, cv2.COLOR_BGR2GRAY)**. Then, we resize the image to 24*24 pixels as our model was trained on 24*24 pixel images **cv2.resize(r_eye, (24,24)).** We normalize our data for better convergence **r_eye = r_eye/255** (All values will be between 0-1). Expand the dimensions to feed into our classifier. We loaded our model using **model = load_model('models/cnnCat2.h5')** . Now we predict each eye with our model **lpred = model.predict_classes(l_eye**). If the value of lpred[0] = 1, it states that eyes are open, if value of lpred[0] = 0 then, it states that eyes are closed.

Sample output:




drowsinessdetection.py - Visual Studio Code
GUI.py
model.py
drowsinessdetection.py
AESE.py
D: > Drowsiness detection > drowsinessdetection.py
n=5
face = cv2.CascadeClassifier(cv2.data.haarcascades+'haarcascade_frontalface_alt.xml')
leye = cv2.CascadeClassifier(cv2.data.haarcascades+'haarcascade_lefteye_2splits.xml')
reye = cv2.CascadeClassifier(cv2.data.haarcascades+'haarcascade_righteye_2splits.xml')
lbl = ['Close', 'Open']
model = load_model('D:\Drowsiness det
path = os.getcwd()
cap = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_COMPLEX_SMALL
count = 0
score = 0
thicc = 2
rpred = 99.0
lpred = 99.0
while (True):
ret, frame = cap.read()
height, width = frame.shape[:2]
gray = cv2.cvtColor(frame, cv2.CO
faces = face.detectMultiScale(gra
left_eye = leye.detectMultiScale(
frame
Closed Score:11
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE
PS C:\Users\saine> python -u "d:\Drowsiness de
1234
PS C:\Users\saine> python -u "d:\Drowsiness de
Traceback (most recent call last):
File "d:\Drowsiness detection\drowsinessdetection.py", line 73, in <module>
if (rpred[0][0]>=0.3 and lpred[0][0]>=0.3):
TypeError: 'int' object is not subscriptable
[WARN:1] global D:\a\opencv-python\opencv-python\opencv\modules\videoio\src\cap_msmf.cpp (438) `anonymous-namespace'::SourceReaderCB::~SourceReaderCB terminating async callback
PS C:\Users\saine> python -u "d:\Drowsiness detection\drowsinessdetection.py"
Ln 26, Col 13
Spaces: 4
UTF-8
CRLF
Python
23°C Mostly cl...
ENG
21:30

Encryption

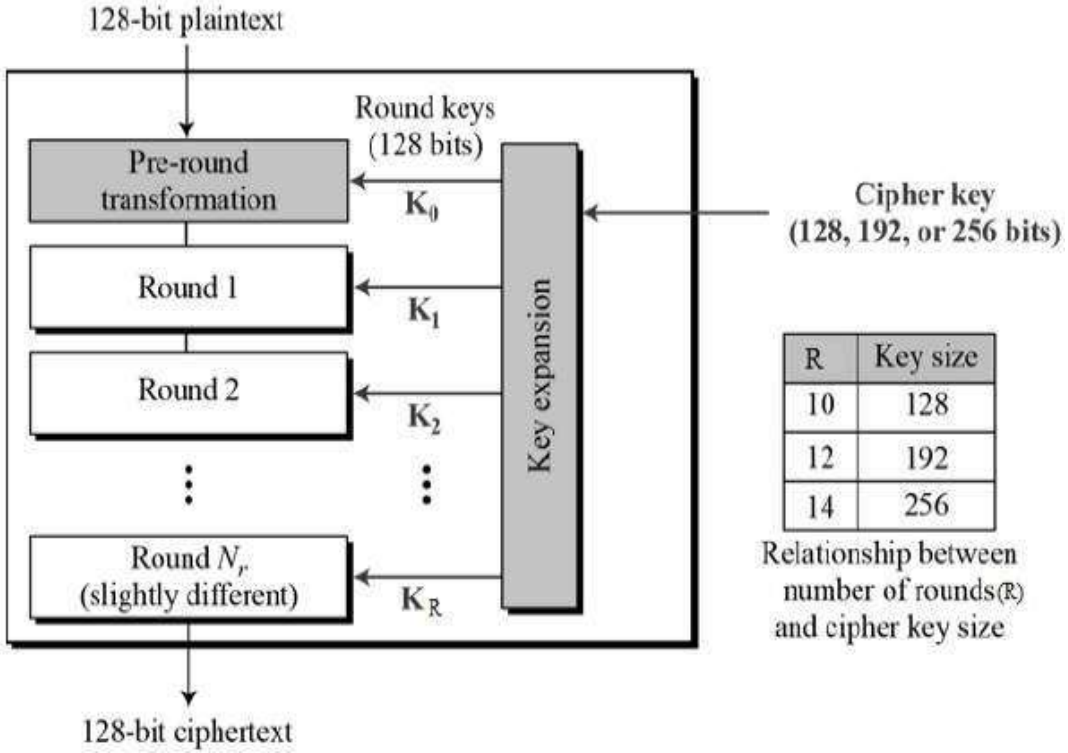
- Encryption is a means of securing digital data using one or more mathematical techniques, along with a password or "key" used to decrypt the information.
- The encryption process translates information using an algorithm that makes the original information unreadable.
- In this project we need encryption to protect private information, sensitive data of users.
- In essence, when your data is encrypted, even if an unauthorized person or entity gains access to the data, they will not be able to read it.
- We use AES encryption in this project.

AES Encryption

- Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the U.S National Institute of Standards and Technology (NIST) in 2001. AES is widely used today as it is a much stronger than DES and triple DES despite being harder to implement.
- It takes 128 bits as input and outputs 128 bits of encrypted cipher text as output. AES relies on substitution-permutation network principle which means it is performed using a series of linked operations which involves replacing and shuffling of the input data.
- For a key length of 128 bits we perform 10 rounds of AES process.

AES Encryption

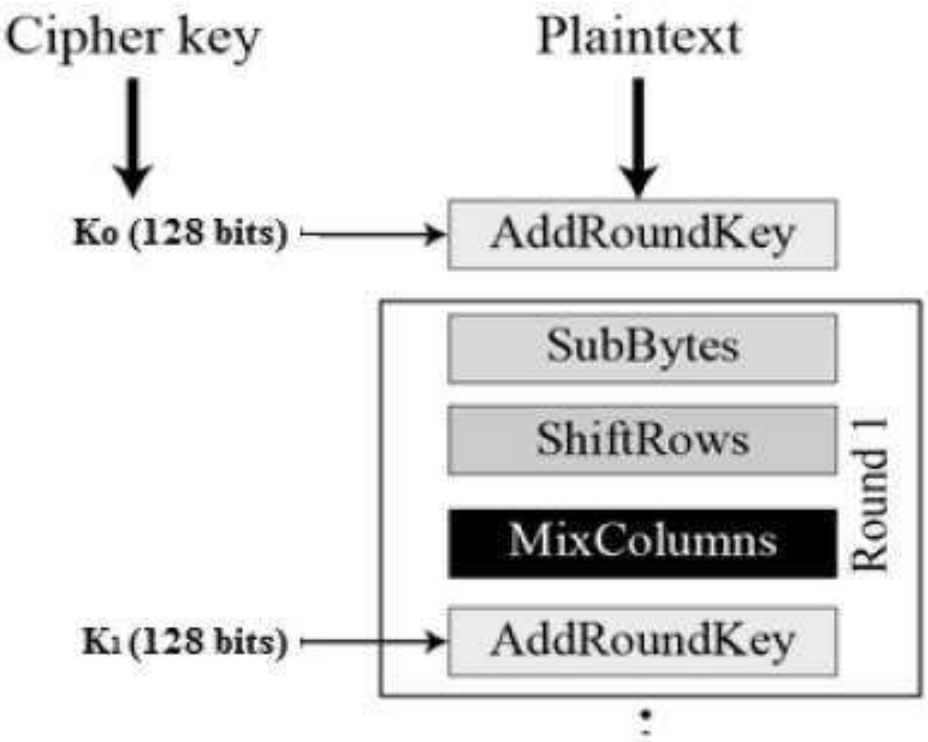
Schematic of AES encryption process



R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

Process in each round



- Plaintext is taken and is processed through the following four steps :

1. Byte Substitution (SubBytes)
2. Shiftrows
3. MixColumns
4. Addroundkey

Byte Substitution (SubBytes)

- The 16 input bytes are substituted by the ones in the S-box. The result is in a matrix of four rows and four columns.

		Y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
X	0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
	4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
	a	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
	b	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
	c	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	d	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
	e	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	f	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

For example :

20 in hex is substituted as B7 according to the table.
In this way all the 16 bytes are substituted

Shiftrows

- Each of the four rows of the matrix is shifted to the left. Any entries that ‘fall off’ are re-inserted on the right side of row. Shift is carried out as follows −

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.
- The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

MixColumns

- Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

X

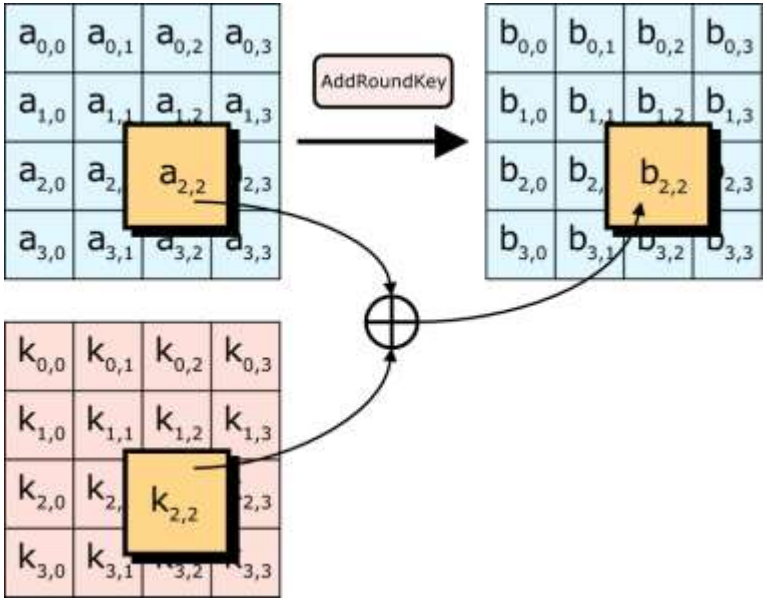
2	3	1	1
1	2	3	1
1	1	2	3
3	1	1	2

=

$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$
$b_{1,0}$	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$
$b_{2,0}$	$b_{2,1}$	$b_{2,2}$	$b_{2,3}$
$b_{3,0}$	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$

Add Round Key

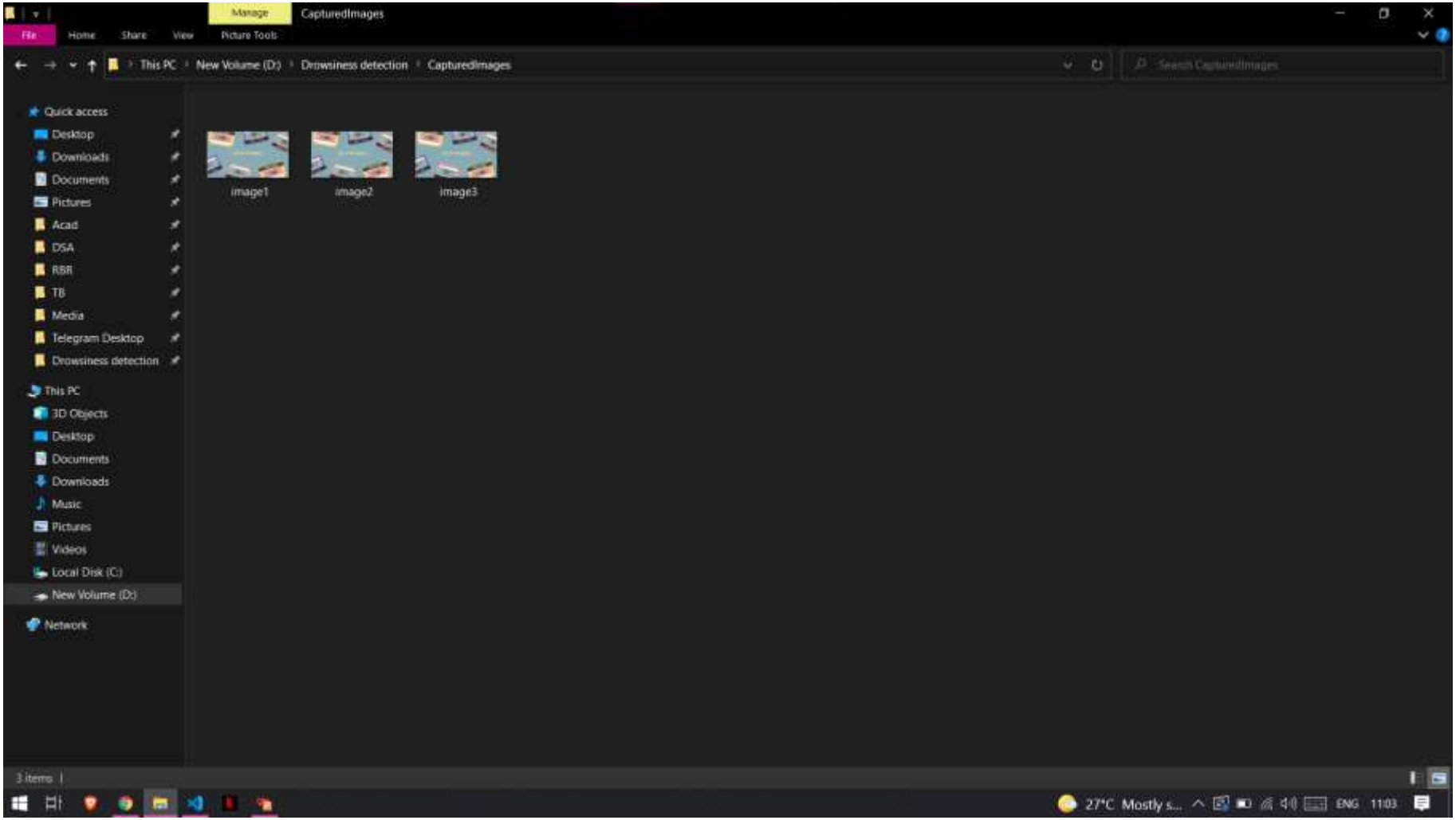
- The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.



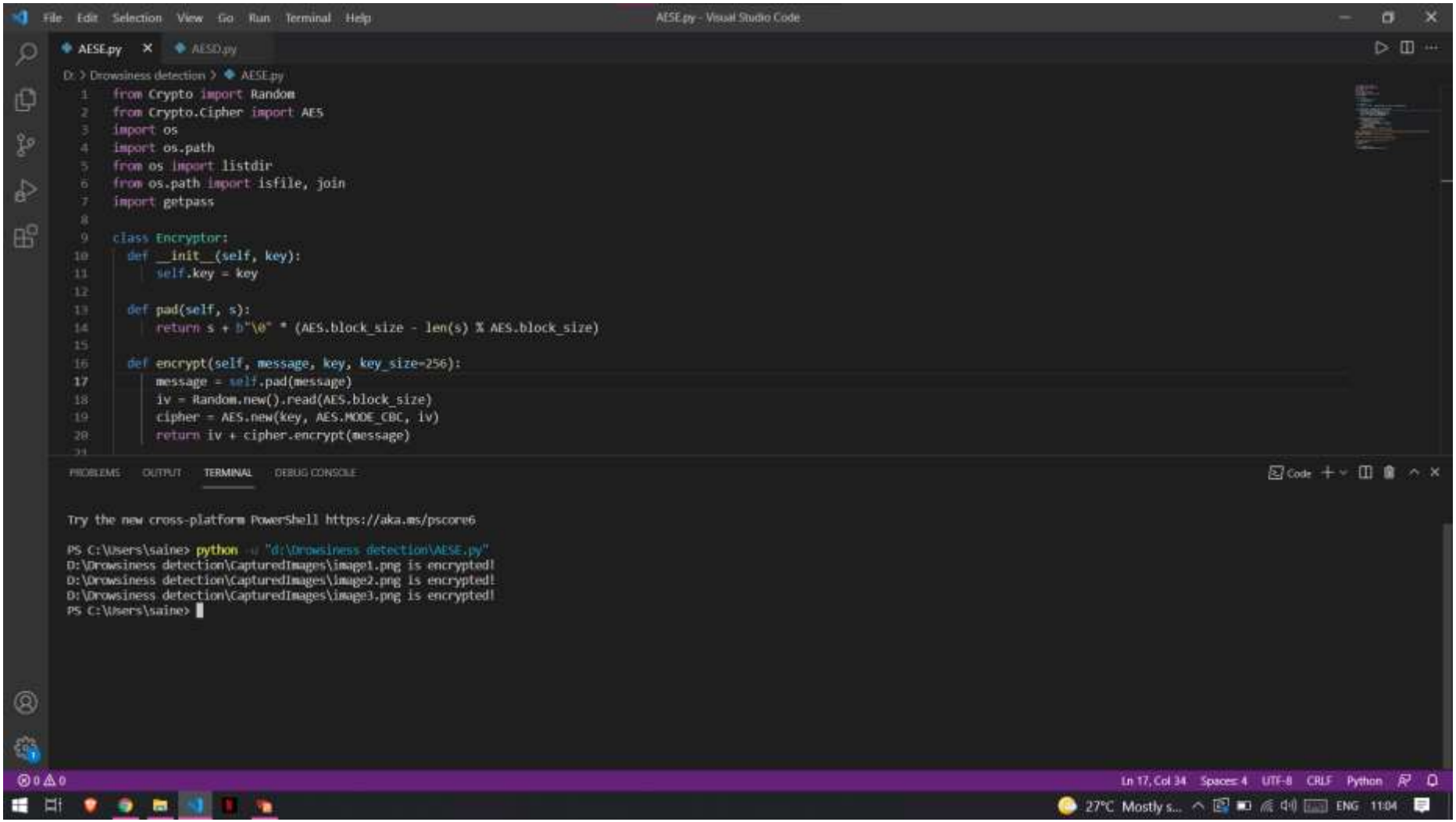
AES Encryption

- Python provides us with a module called Crypto in which we have libraries like AES, Random etc.
- In this project, we considered a key of length 32 bytes or 256 bits.
- So we perform the 14 rounds of AES process.
- In this project we operate in CBC mode.
- CBC is a mode of operation where each plaintext block gets XOR-ed with the previous ciphertext block prior to encryption.
- It is one of the most commonly used block cipher modes of operation due to its ease of implementation and support for parallelized decryption.

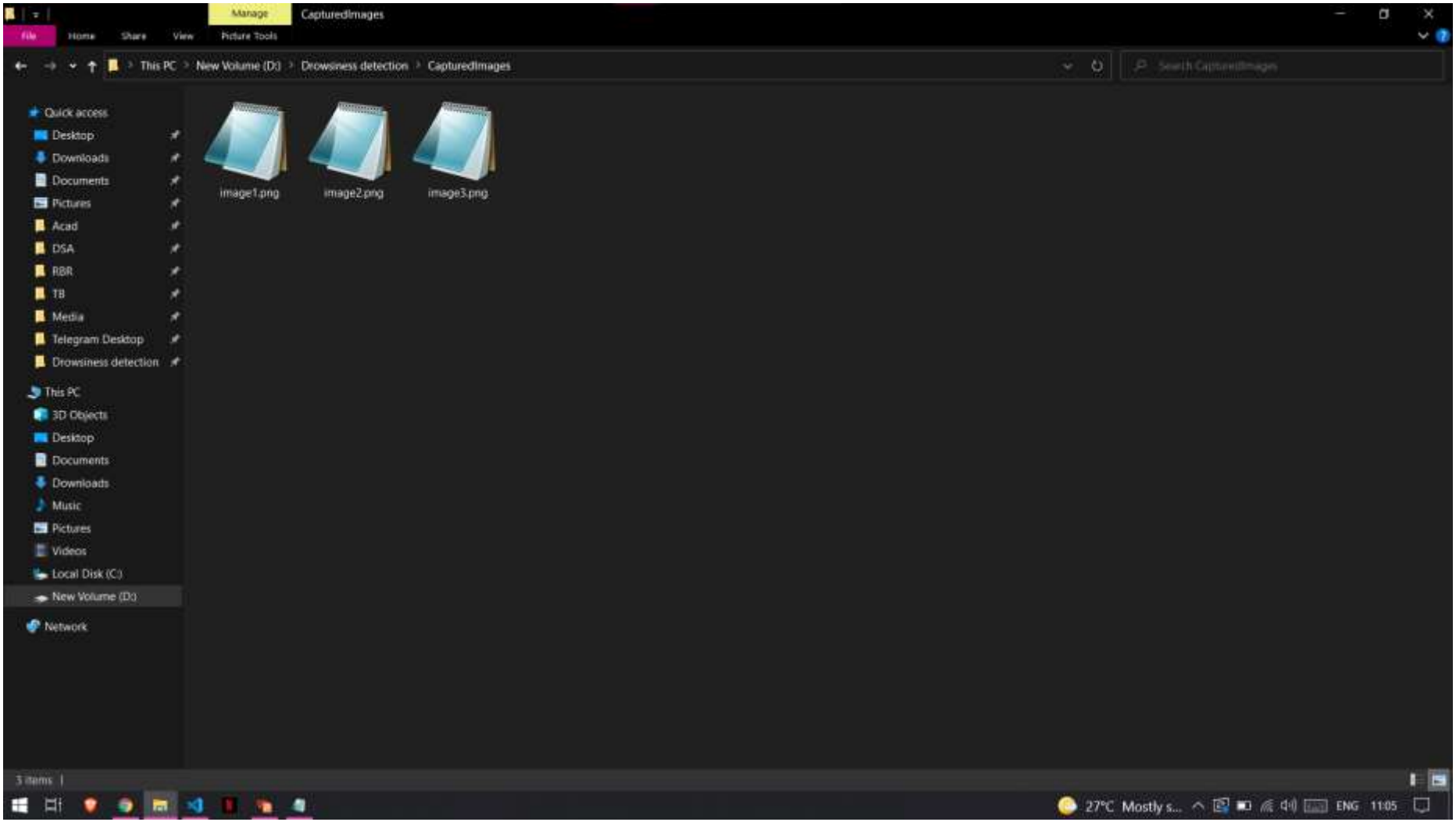
Sample images in the directory



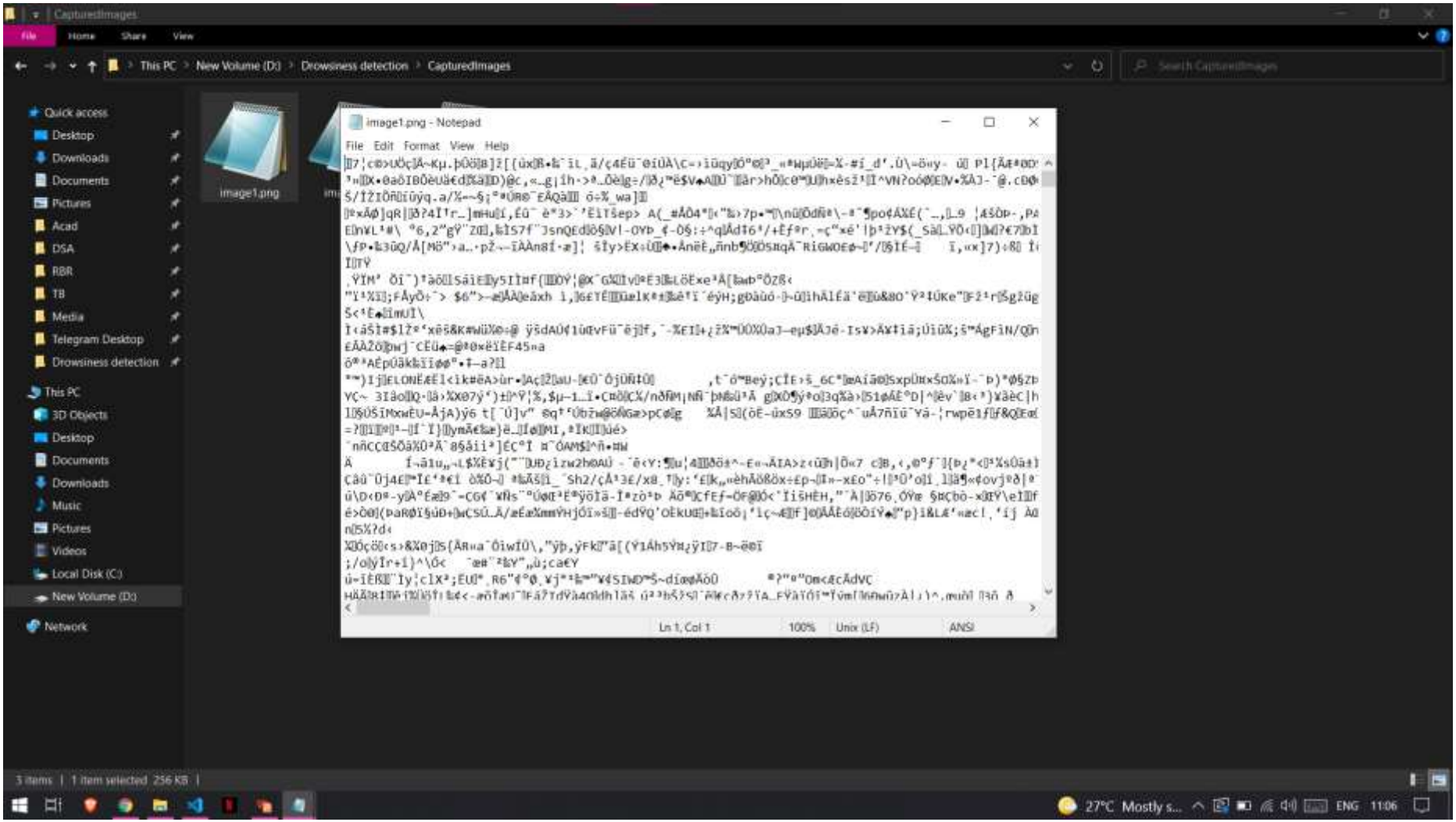
Encryption


File Edit Selection View Go Run Terminal Help
AESE.py - Visual Studio Code
AESE.py
AESD.py
D: > Drowsiness detection > AESE.py
from Crypto import Random
from Crypto.Cipher import AES
import os
import os.path
from os import listdir
from os.path import isfile, join
import getpass
class Encryptor:
def __init__(self, key):
self.key = key
def pad(self, s):
return s + b"\0" * (AES.block_size - len(s) % AES.block_size)
def encrypt(self, message, key, key_size=256):
message = self.pad(message)
iv = Random.new().read(AES.block_size)
cipher = AES.new(key, AES.MODE_CBC, iv)
return iv + cipher.encrypt(message)
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
Code
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\saine> python -u "d:\Drowsiness detection\AESE.py"
D:\Drowsiness detection\CapturedImages\image1.png is encrypted!
D:\Drowsiness detection\CapturedImages\image2.png is encrypted!
D:\Drowsiness detection\CapturedImages\image3.png is encrypted!
PS C:\Users\saine>
Ln 17, Col 34 Spaces: 4 UTF-8 CRLF Python
27°C Mostly s... ENG 11:04

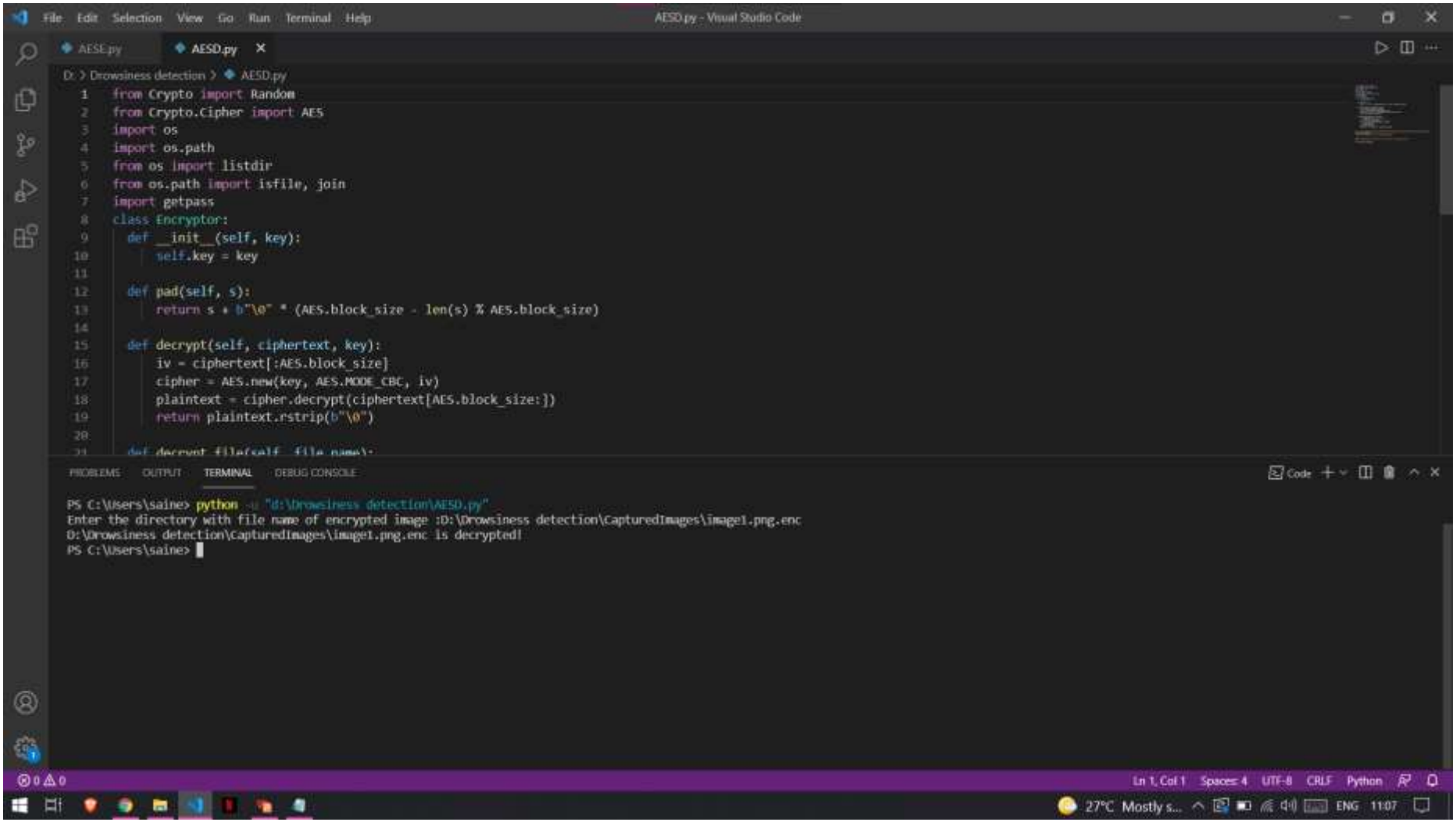
Sample images after Encryption



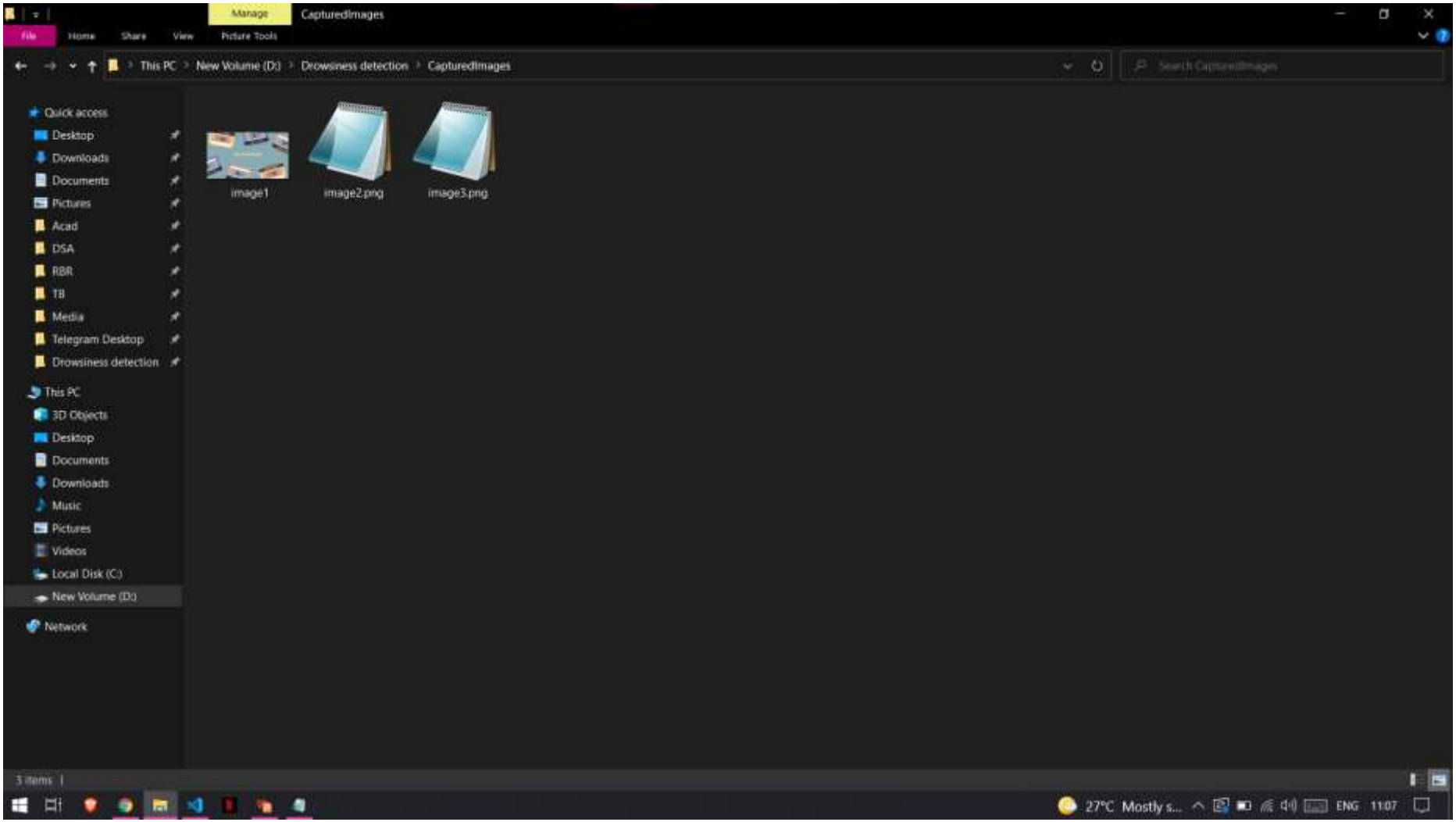
Unreadable format



Decryption


File Edit Selection View Go Run Terminal Help
AESD.py - Visual Studio Code
AESE.py
AESD.py
D: > Drowsiness detection > AESD.py
1 from Crypto import Random
2 from Crypto.Cipher import AES
3 import os
4 import os.path
5 from os import listdir
6 from os.path import isfile, join
7 import getpass
8 class Encryptor:
9 def __init__(self, key):
10 self.key = key
11
12 def pad(self, s):
13 return s + b"\0" * (AES.block_size - len(s) % AES.block_size)
14
15 def decrypt(self, ciphertext, key):
16 iv = ciphertext[:AES.block_size]
17 cipher = AES.new(key, AES.MODE_CBC, iv)
18 plaintext = cipher.decrypt(ciphertext[AES.block_size:])
19 return plaintext.rstrip(b"\0")
20
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
Code
PS C:\Users\saine> python -u "d:\Drowsiness detection\AESD.py"
Enter the directory with file name of encrypted image :D:\Drowsiness detection\CapturedImages\image1.png.enc
D:\Drowsiness detection\CapturedImages\image1.png.enc is decrypted!
PS C:\Users\saine>
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python
27°C Mostly s... ENG 11:07

Decrypted files



References

1. A journal on Driver Drowsiness Detection by V B Navya Kiran, Raksha R and team published on IJERT

2. A journal on Image encryption and decryption using AES algorithm by Priya Deshmukh published on IJSER

THANK YOU!