**Drowsiness Detection System**

What

A Drowsiness Detection System is a safety system—often used in cars or heavy machinery—that detects when a driver is becoming drowsy or falling asleep. It uses various technologies like cameras.

It is a computer vision application which can detect whether a person is sleeping or not and then store their data who are sleeping.

This is a web application which can be accessed over a **LAN.**It is a CNN model.

Why

**Road Safety**: Drowsy driving is a major cause of road accidents.

**Early Warning**: It alerts the driver before they fall asleep, giving them time to react.

**Regulatory Compliance**: In industries like trucking, such systems help meet safety regulations.

How

**1-Backend**

Connection with IP Camera : OpenCV

Drowsy Detection : OpenCV

Non Drowsy Detection : Keras,Numpy

Save Data : OpenCV

**2-Frontend**

Web Application : Streamlit

#### Camera-based (Most Common)

* Monitors facial features like **eye blinking rate**, **eye closure duration**
* Uses computer vision and AI models to detect signs of drowsiness.
* Example tools: OpenCV, Deep Learning models (CNN, etc.)

**CODE FOR CAPTURING VIDEOS WITH HELP OF OPENCV**

**import cv2**

**vid=cv2.VideoCapture("video1.mp4")**

**facemodel=cv2.CascadeClassifier("face.xml")**

**while(vid.isOpened()):**

**flag,frame=vid.read()**

**if(flag):**

**face=facemodel.detectMultiScale(frame)#[[x,y,l,w],[]]**

**for(x,y,l,w) in face:**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,0,0),2)**

**cv2.namedWindow("Drowiess Detection System",cv2.WINDOW\_NORMAL)**

**cv2.imshow("Drowiess Detection System",frame)**

**key=cv2.waitKey(20)**

**if(key==ord("x")):**

**break**

**else:**

**break**

**vid.release**

**cv2.destroyAllWindows()**

**CODE FOR TRANINNG THE MODEL**

**from keras.layers import Conv2D,MaxPooling2D,Flatten,Dense**

**from keras.models import Sequential**

**from tensorflow.keras.optimizers import Adam**

**from tensorflow.keras.preprocessing.image import ImageDataGenerator**

**#define the model**

**mymodel=Sequential()**

**mymodel.add(Conv2D(32,(3,3),activation='relu',input\_shape=(150,150,3)))**

**mymodel.add(MaxPooling2D())**

**mymodel.add(Conv2D(32,(3,3),activation='relu'))**

**mymodel.add(MaxPooling2D())**

**mymodel.add(Conv2D(32,(3,3),activation="relu"))**

**mymodel.add(MaxPooling2D())**

**mymodel.add(Flatten())**

**mymodel.add(Dense(100,activation="relu"))**

**mymodel.add(Dense(1,activation="sigmoid"))**

**mymodel.compile(optimizer="Adam",loss="binary\_crossentropy",metrics=["accuracy"])**

**#define the data**

**train=ImageDataGenerator(rescale=1./255,shear\_range=0.2,zoom\_range=0.2,horizontal\_flip="True")**

**test=ImageDataGenerator(rescale=1./255)**

**train\_img=train.flow\_from\_directory("train",target\_size=(150,150),batch\_size=16,class\_mode="binary")**

**test\_img=test.flow\_from\_directory("test",target\_size=(150,150),batch\_size=16,class\_mode="binary")**

**#train and test model**

**drowiess\_model=mymodel.fit(train\_img,epochs=10,validation\_data=test\_img)**

**#save**

**mymodel.save("drowijess.h5",drowiess\_model)**

**CODE FOR CAPTURING IMAGES WITH HELP OF OPENCV AND DETECTING THE DROWSINESS USING CNN MODEL.**

**import cv2**

**from keras.models import load\_model**

**from keras.utils import load\_img,img\_to\_array**

**import numpy as np**

**#**

**facemodel=cv2.CascadeClassifier("face.xml")**

**dds=load\_model("drowiess.h5")**

**img=cv2.imread("open.png")**

**face=facemodel.detectMultiScale(img)**

**for (x,y,l,w) in face:**

**crop\_face=img[y:y+w,x:x+l]**

**cv2.imwrite("temp.jpg",crop\_face)**

**crop\_face=load\_img("temp.jpg",target\_size=(150,150,3))**

**crop\_face=np.expand\_dims(crop\_face,axis=0)**

**pred=dds.predict(crop\_face)[0][0]**

**if pred==1:**

**cv2.rectangle(img,(x,y),(x+l,y+w),(0,255,0),4)**

**else:**

**cv2.rectangle(img,(x,y),(x+l,y+w),(0,0,255),4)**

**cv2.namedWindow("dds",cv2.WINDOW\_NORMAL)**

**cv2.imshow("dds",img)**

**cv2.waitKey(10000)**

**cv2.destroyAllWindows()**

**CODE FOR CAPTURING VIDEOS WITH HELP OF OPENCV AND DETECTING THE DROWSINESS USING CNN MODEL.**

**import cv2**

**from keras.models import load\_model**

**from keras.utils import load\_img,img\_to\_array**

**import numpy as np**

**facemodel=cv2.CascadeClassifier("face.xml")**

**maskmodel=load\_model("drowiess.h5")**

**vid=cv2.VideoCapture("video1.mp4")**

**i=1**

**while(vid.isOpened()):**

**flag,frame=vid.read()**

**if(flag):**

**face=facemodel.detectMultiScale(frame)**

**for (x,y,l,w) in face :**

**crop\_face1=frame[y:y+w,x:x+l]**

**crop\_face=cv2.imwrite("temp.jpg",crop\_face1)**

**crop\_face=load\_img("temp.jpg",target\_size=(150,150))**

**crop\_face=img\_to\_array(crop\_face)**

**crop\_face=np.expand\_dims(crop\_face,axis=0)**

**pred=maskmodel.predict(crop\_face)[0][0]**

**if(pred==1):**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,255,0),3)**

**print("Active")**

**else:**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,0,255),3)**

**cv2.namedWindow("adil",cv2.WINDOW\_NORMAL)**

**cv2.imshow("adil",frame)**

**key=cv2.waitKey(15)**

**if(key==ord("x")):**

**break**

**else:**

**break**

**cv2.destroyAllWindows()**

**COMPLETE CODE FOR DROWSINESS DETECTION SYSTEM USING STREAMLIT FOR HOSTING ON LAN NETWORK**

**import streamlit as st**

**import numpy as np**

**import cv2**

**import tempfile**

**from keras.models import load\_model**

**from keras.utils import load\_img,img\_to\_array**

**facemodel=cv2.CascadeClassifier("face.xml")**

**model=load\_model('drowiess.h5')**

**st.title("Drowsiness Detection System ")**

**choice=st.sidebar.selectbox("MY MENU",("HOME","IMAGE","VIDEO","WEB CAMERA","URL"))**

**if(choice=="HOME"):**

**st.header("Welcome to Drowsiness Detection application")**

**st.image("https://www.mdpi.com/applsci/applsci-11-08441/article\_deploy/html/images/applsci-11-08441-g001.png")**

**elif(choice=="IMAGE"):**

**file=st.file\_uploader("upload image")**

**if file:**

**b=file.getvalue()**

**d=np.frombuffer(b,np.uint8)**

**frame=cv2.imdecode(d,cv2.IMREAD\_COLOR)**

**face=facemodel.detectMultiScale(frame)**

**for (x,y,l,w) in face :**

**crop\_face1=frame[y:y+w,x:x+l]**

**crop\_face=cv2.imwrite("temp.jpg",crop\_face1)**

**crop\_face=load\_img("temp.jpg",target\_size=(150,150))**

**crop\_face=img\_to\_array(crop\_face)**

**crop\_face=np.expand\_dims(crop\_face,axis=0)**

**pred=model.predict(crop\_face)[0][0]**

**if(pred==1):**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,255,0),3)**

**else:**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,0,255),3)**

**st.image(frame,channels="BGR",width=400)**

**elif(choice=="VIDEO"):**

**file=st.file\_uploader("upload video")**

**window=st.empty()**

**if file:**

**tfile=tempfile.NamedTemporaryFile()**

**tfile.write(file.read())**

**vid=cv2.VideoCapture(tfile.name)**

**i=1**

**while(vid.isOpened()):**

**flag,frame=vid.read()**

**if(flag):**

**face=facemodel.detectMultiScale(frame)**

**for (x,y,l,w) in face :**

**crop\_face1=frame[y:y+w,x:x+l]**

**crop\_face=cv2.imwrite("temp.jpg",crop\_face1)**

**crop\_face=load\_img("temp.jpg",target\_size=(150,150))**

**crop\_face=img\_to\_array(crop\_face)**

**crop\_face=np.expand\_dims(crop\_face,axis=0)**

**pred=model.predict(crop\_face)[0][0]**

**if(pred==1):**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,255,0),3)**

**path="C:/drowies/dds/Scripts/DATA/EYESCLOSED/"+"str(i)"+".jpg"**

**cv2.imwrite(path,crop\_face)**

**i=i+1**

**else:**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,0,255))**

**window.image(frame,channels="BGR")**

**elif(choice=="WEB CAMERA"):**

**btn=st.button("start camera")**

**window=st.empty()**

**btn2=st.button("stop camera")**

**if btn2:**

**st.rerun()**

**if btn:**

**vid=cv2.VideoCapture(0)**

**while(vid.isOpened()):**

**flag,frame=vid.read()**

**if(flag):**

**model=load\_model('drowiess.h5')**

**face=facemodel.detectMultiScale(frame)**

**for (x,y,l,w) in face :**

**crop\_face1=frame[y:y+w,x:x+l]**

**crop\_face=cv2.imwrite("temp.jpg",crop\_face1)**

**crop\_face=load\_img("temp.jpg",target\_size=(150,150))**

**crop\_face=img\_to\_array(crop\_face)**

**crop\_face=np.expand\_dims(crop\_face,axis=0)**

**pred=model.predict(crop\_face)[0][0]**

**if(pred==1):**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,255,0),3)**

**else:**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,0,255),3)**

**window.image(frame,channels="BGR")**

**btn2=st.button("stop camera")**

**elif(choice=="URL"):**

**a=st.text\_input("enter the url ")**

**btn=st.button("start camera")**

**window=st.empty()**

**if btn:**

**vid=cv2.VideoCapture(a)**

**i=1**

**while(vid.isOpened()):**

**flag,frame=vid.read()**

**if(flag):**

**face=facemodel.detectMultiScale(frame)**

**for (x,y,l,w) in face :**

**crop\_face1=frame[y:y+w,x:x+l]**

**crop\_face=cv2.imwrite("temp.jpg",crop\_face1)**

**crop\_face=load\_img("temp.jpg",target\_size=(150,150))**

**crop\_face=img\_to\_array(crop\_face)**

**crop\_face=np.expand\_dims(crop\_face,axis=0)**

**pred=model.predict(crop\_face)[0][0]**

**if(pred==1):**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,255,0),3)**

**else:**

**cv2.rectangle(frame,(x,y),(x+l,y+w),(0,0,255),3)**

**window.image(frame,channels="BGR")**

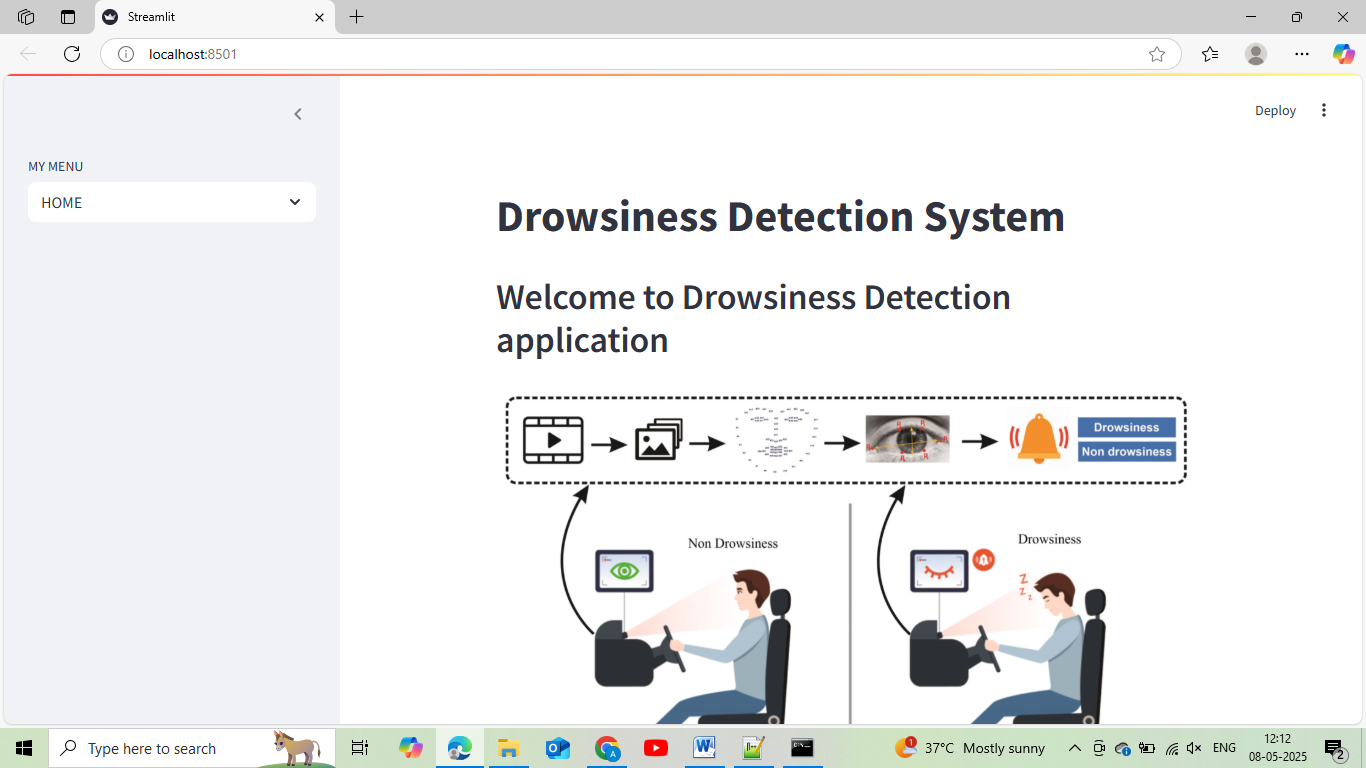
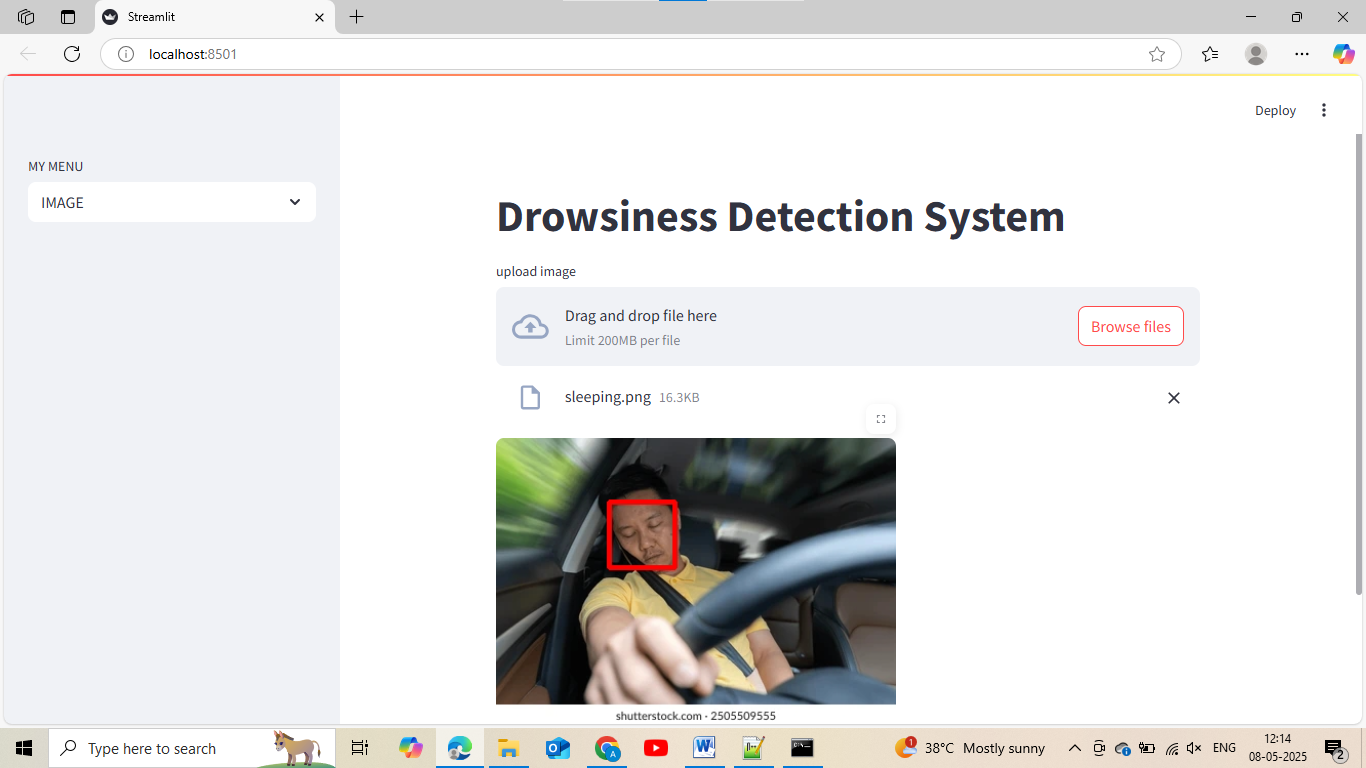
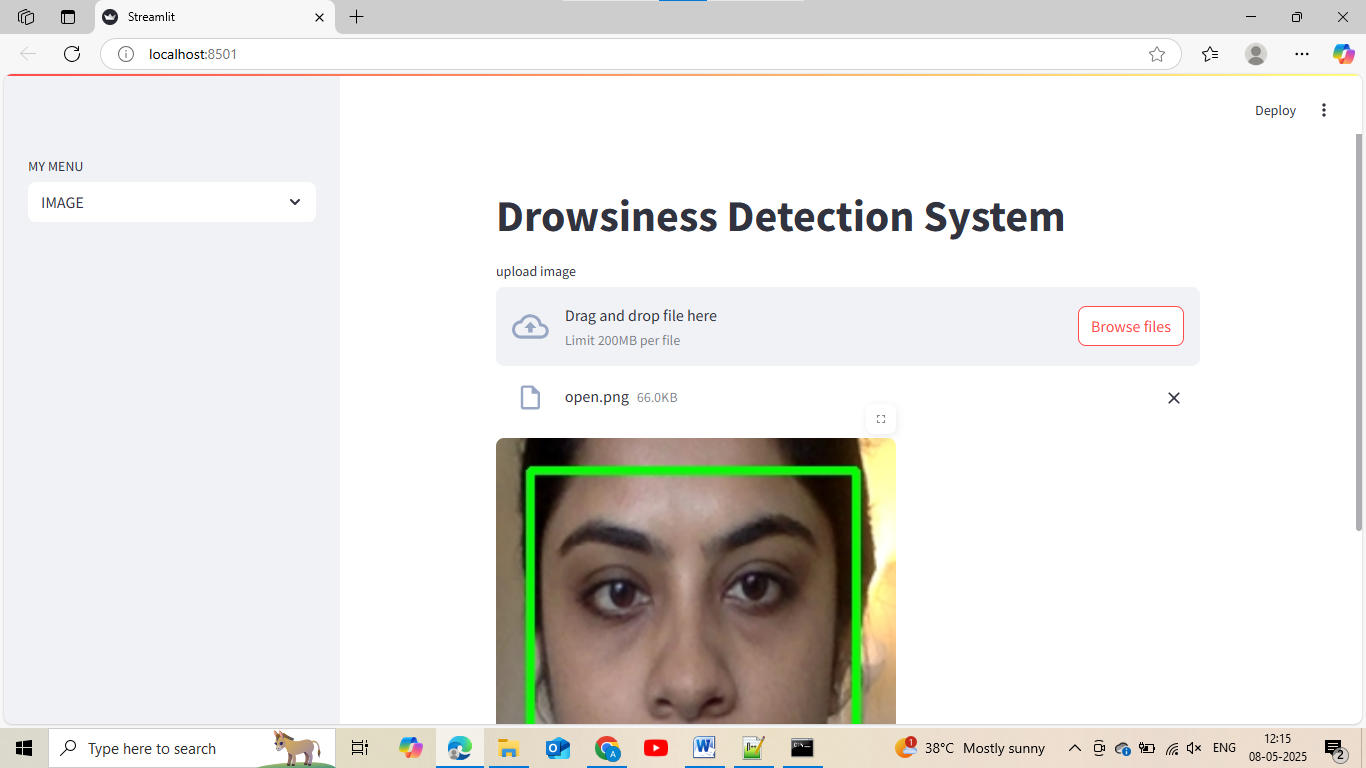
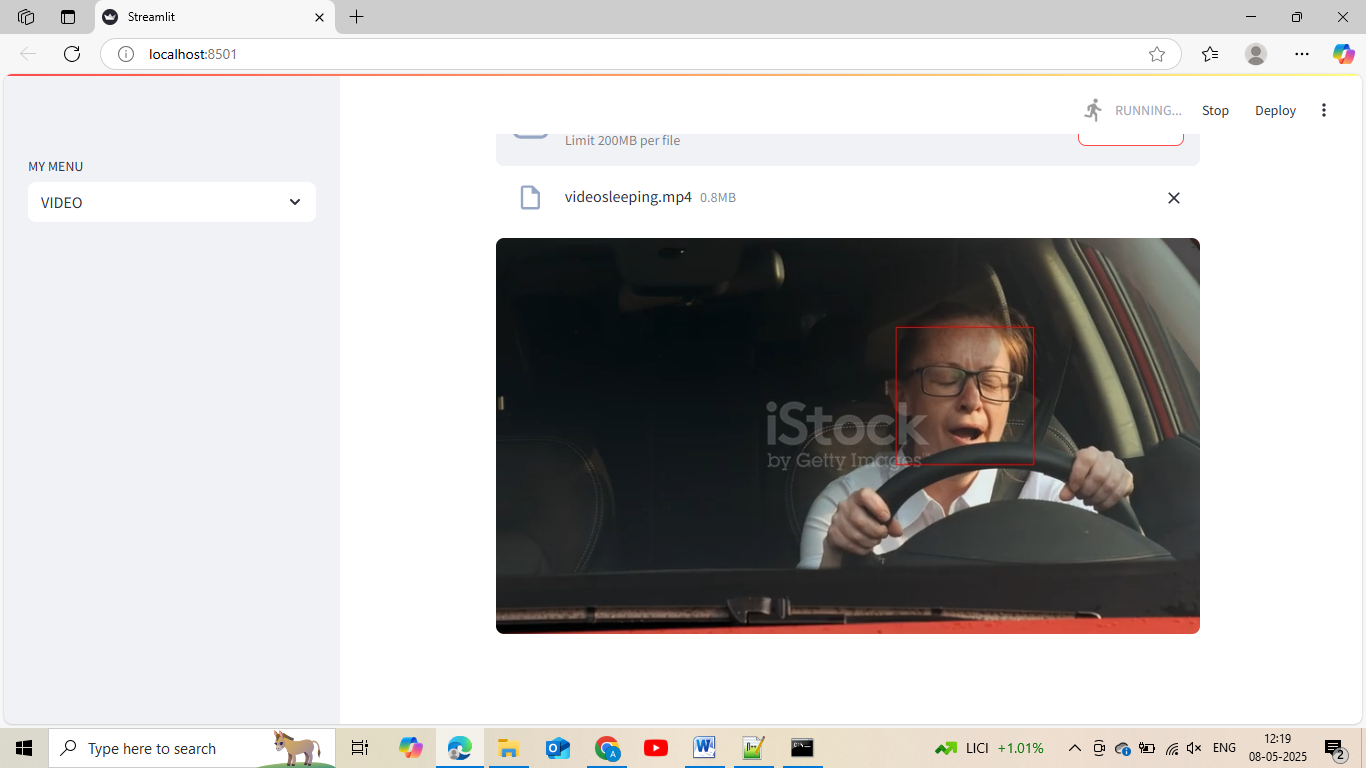
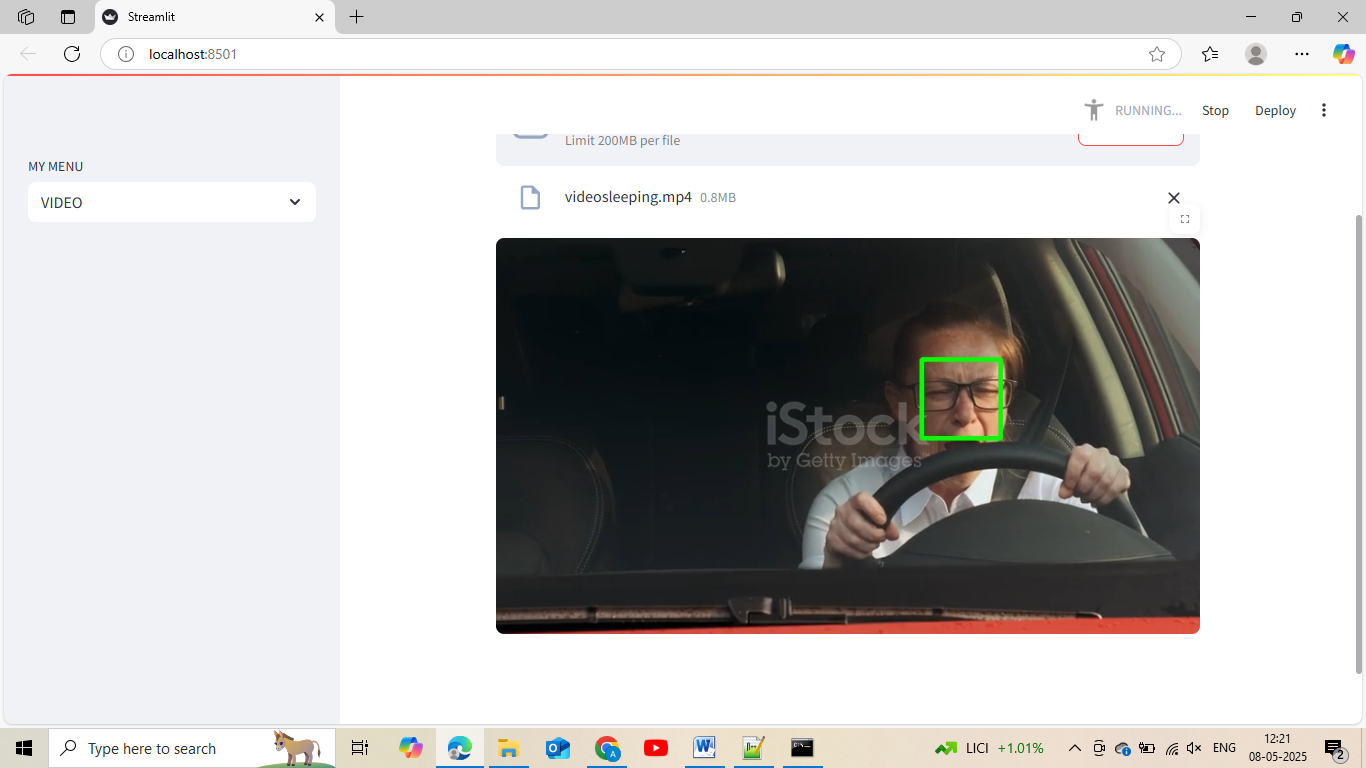
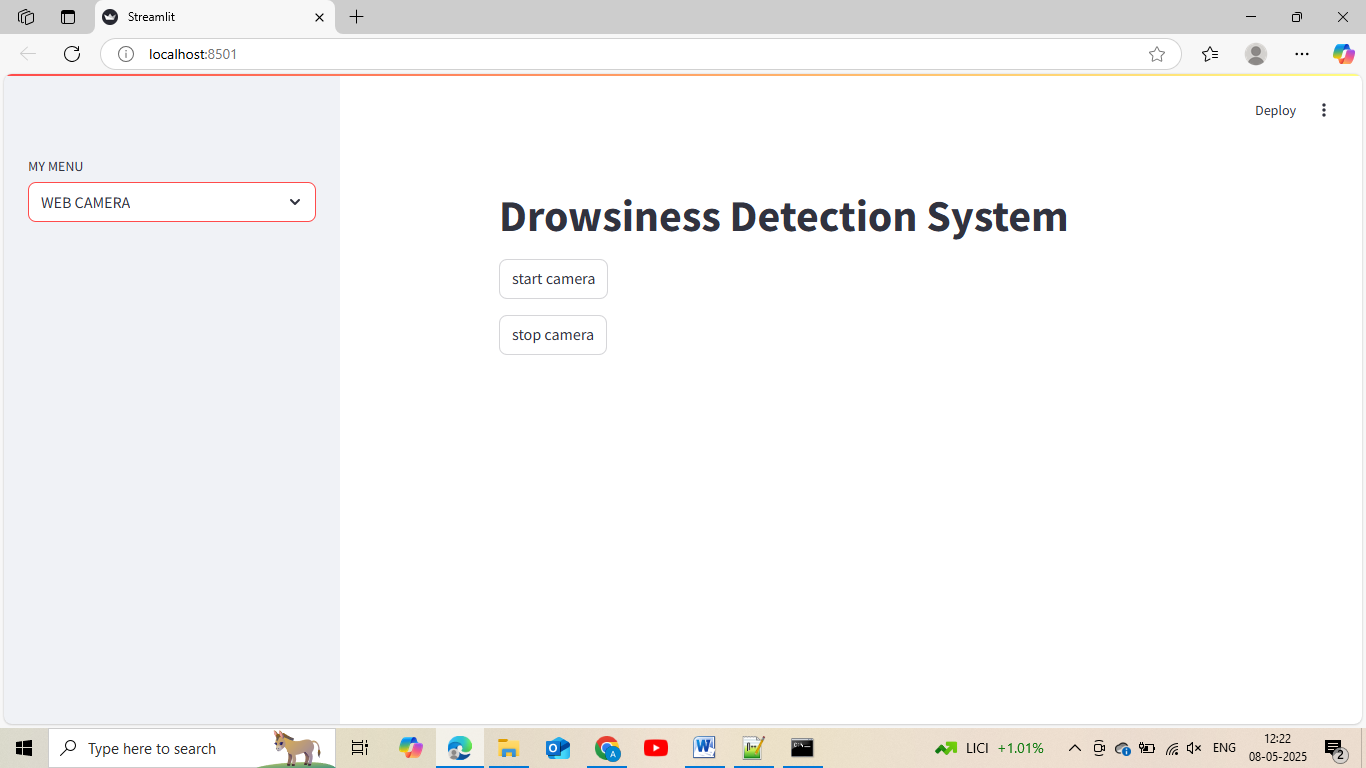
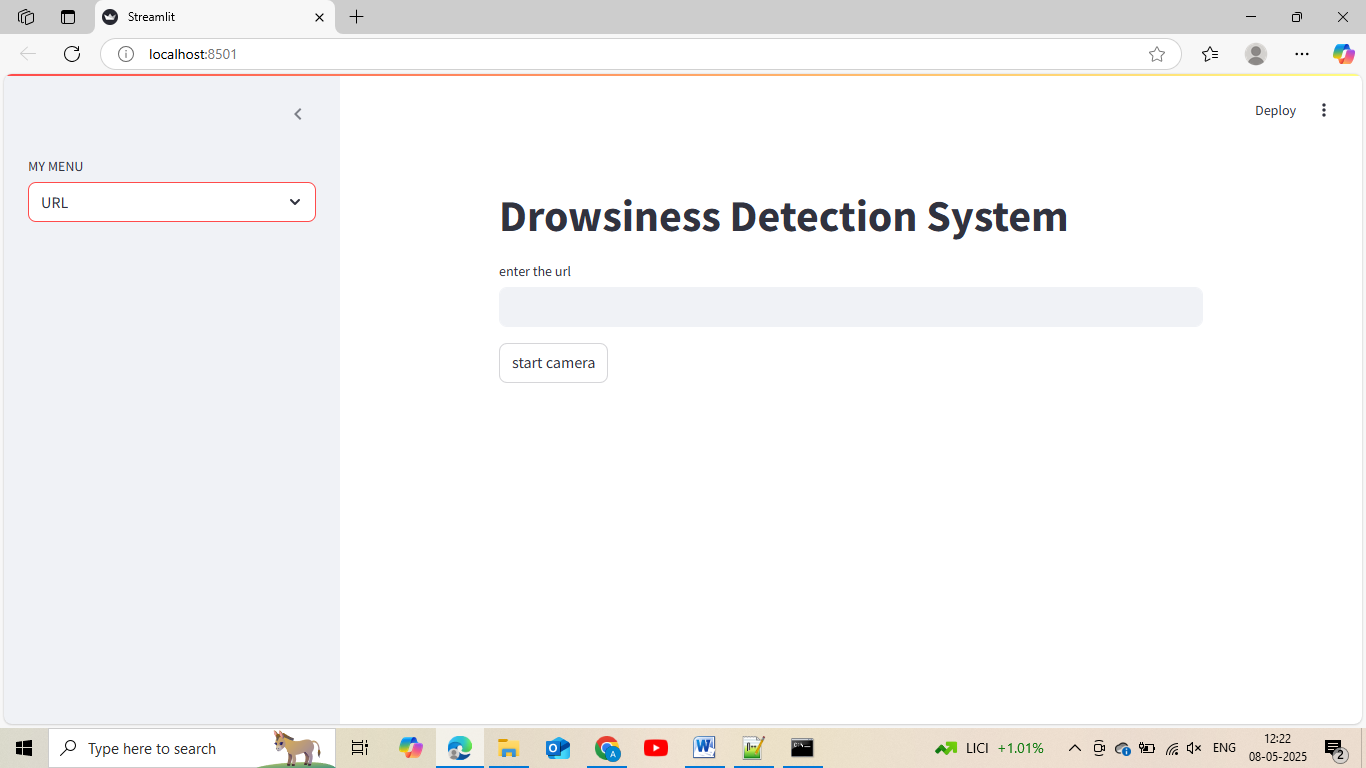
**btn2=st.button("stop camera")**

**if btn2:**

**vid.close()**

**st.experimental\_rerun()**

**SCREENSHOT FRONTEND DESIGN OF DROWSINESS DETECTION SYSTEM**

**** ****     

**LIBRARY USED IN THIS PROJECT**

**pip install opencv-python**

**pip install tensorflow , pip install imutils , pip install tempfile**

**pip install numpy , pip install numpy**