DRIVER DROWSINESS DETECTION SYSTEM🚗

WHAT:

It is a computer vision applicatio n which can detect whether the eyes are open or not and store the data whose eyes are not open

This application can get data from various sources such as IP cam,webcam,online videos

This is a web application which can be accessed over LAN

WHY:

Implementing a driver drowsiness detection system addresses critical safety concerns, enhances overall transportation efficiency, and aligns with advancements in technology for the improvement of road safety and driver well-being

This application can be used in Automotive Industry,Public Transportation and Research & Studies

HOW:

Backend:Connection with IP Camera:OpenCV

Face detection:OpenCV

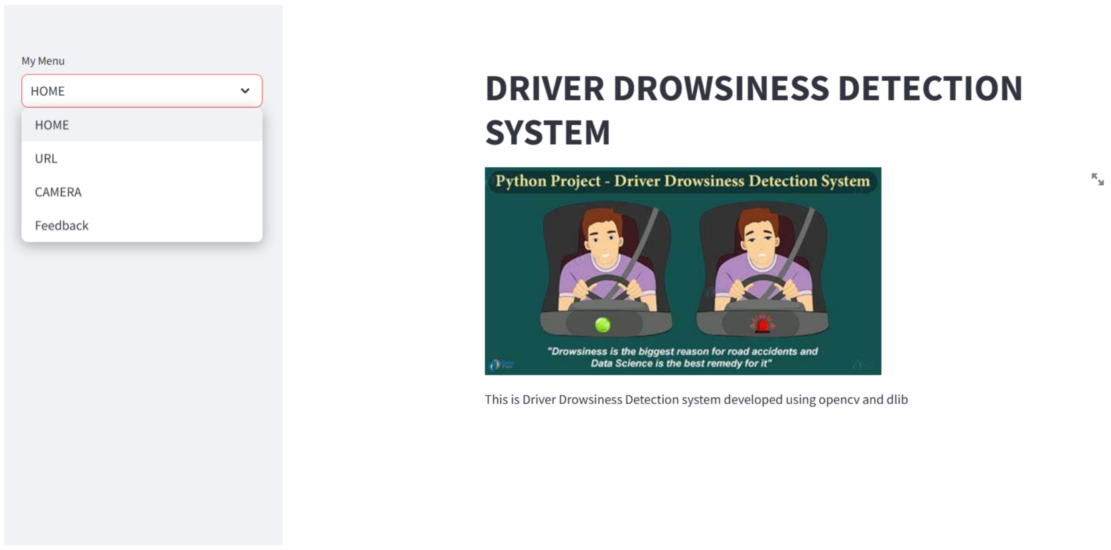
Eye detection:PIL

Save data:OpenCV

Frontend:Streamlit

1. Introduction:

The Driver Drowsiness Detection System is a computer vision application designed to detect the state of a driver's eyes and store data when eyes are determined to be closed. The system is capable of obtaining data from various sources, including IP cameras, webcams, and online videos. It is implemented as a web application accessible over a Local Area Network (LAN).


My Menu
HOME
HOME
URL
CAMERA
Feedback
Python Project - Driver Drowsiness Detection System
"Drowsiness is the biggest reason for road accidents and Data Science is the best remedy for it"

DRIVER DROWSINESS DETECTION SYSTEM

This is Driver Drowsiness Detection system developed using opencv and dlib

2. Objectives:

- **Eye State Detection:** Determine whether the driver's eyes are open or closed using computer vision techniques.
- **Data Storage:** Save data when closed eyes are detected for further analysis or intervention.
- **Multiple Input Sources:** Support data input from diverse sources such as IP cameras, webcams, and online videos.
- **Web-Based Access:** Enable users to access the application through a web interface over a LAN.

3. Justification:

Implementing a driver drowsiness detection system is essential for several reasons:

- **Safety Enhancement:** Address critical safety concerns by preventing accidents caused by drowsy driving.
- **Efficiency Improvement:** Enhance overall transportation efficiency by ensuring drivers are alert and focused.
- **Regulatory Compliance:** Meet regulatory requirements in regions mandating driver monitoring systems for commercial vehicles.
- **Insurance Incentives:** Potentially reduce insurance premiums by demonstrating a commitment to safety.

4. Applications:

The Driver Drowsiness Detection System finds applications in various sectors, including:

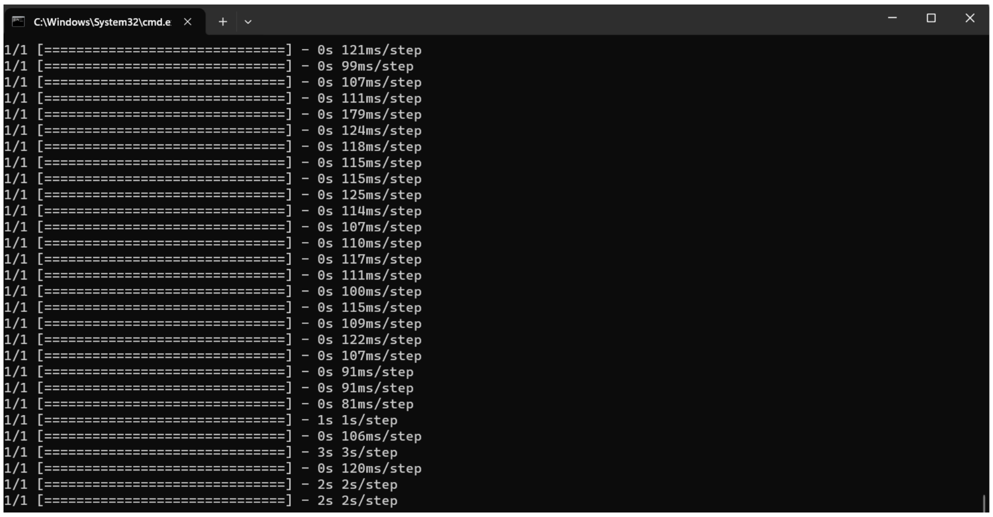
- **Automotive Industry:** Integration into vehicles for real-time driver monitoring.
- **Public Transportation:** Monitoring the alertness of drivers in buses, trains, and other public transport.
- **Research and Studies:** Providing valuable data for research on driver behavior and fatigue patterns.

CODE:

Backend.py

```
import cv2

from keras.models import load_model

from keras.models import load_model

from PIL import Image, ImageOps

import numpy as np

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

eyemodel=load_model("eyes.h5",compile=False)

vid=cv2.VideoCapture("group.mp4")

while(vid.isOpened()):

  flag,frame=vid.read()

  if(flag):

    faces=facemodel.detectMultiScale(frame)

    for(x,y,w,h) in faces:

      face_img=frame[y:y+h,x:x+w]

      size = (224, 224)

      face_img = ImageOps.fit(Image.fromarray(face_img), size, Image.LANCZOS)

      face_img = (np.asarray(face_img).astype(np.float32) / 127.5) - 1

      face_img=np.expand_dims(face_img,axis=0)

      pred=eyemodel.predict(face_img)[0][0]

      if(pred>0.9):

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

      else:

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

    cv2.namedWindow("Havya window",cv2.WINDOW_NORMAL)

    cv2.imshow("Havya window",frame)
```

```
        key=cv2.waitKey(10)

        if(key==ord('x')):

            break

    else:

        break

vid.release()

cv2.destroyAllWindows()
```



```
1/1 [==============================] - 0s 121ms/step
1/1 [==============================] - 0s 99ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 111ms/step
1/1 [==============================] - 0s 179ms/step
1/1 [==============================] - 0s 124ms/step
1/1 [==============================] - 0s 118ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 125ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 110ms/step
1/1 [==============================] - 0s 117ms/step
1/1 [==============================] - 0s 111ms/step
1/1 [==============================] - 0s 100ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 122ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 91ms/step
1/1 [==============================] - 0s 91ms/step
1/1 [==============================] - 0s 81ms/step
1/1 [==============================] - 1s 1s/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 3s 3s/step
1/1 [==============================] - 0s 120ms/step
1/1 [==============================] - 2s 2s/step
1/1 [==============================] - 2s 2s/step
```

<u>Main.py</u>

```
import streamlit as st

import cv2

from keras.models import load_model

from PIL import Image, ImageOps

import numpy as np

st.set_page_config(page_title="DRIVER DROWSINESS DETECTION SYSTEM",page_icon="https://33.media.tumblr.com/5c79953db232e69e2f07f58b0a25c70f/tumblr_ncq090P25b1tlnptjo1_1280.gif")

st.title("DRIVER DROWSINESS DETECTION SYSTEM")

choice=st.sidebar.selectbox("My Menu",("HOME","URL","CAMERA","Feedback"))

if(choice=="HOME"):
    st.image("https://th.bing.com/th/id/OIP.gC1o75jJN-xJLKr8B0hiQwHaD4?
rs=1&pid=ImgDetMain")

    st.write("This is Driver Drowsiness Detection system developed using opencv and dlib")

elif(choice=="URL"):

    url=st.text_input("Enter the url")

    btn=st.button("Start Detection")

    window=st.empty()

    if btn:

        i=1

        btn2=st.button('Stop detection')

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

        eyemodel=load_model("eyes.h5",compile=False)

        vid=cv2.VideoCapture(url)

        if btn2:

            vid.release()

            st.experimental_rerun()

        while(vid.isOpened()):

            flag,frame=vid.read()

            if(flag):

                faces=facemodel.detectMultiScale(frame)

                for(x,y,w,h) in faces:

                    face_img=frame[y:y+h,x:x+w]

                    size = (224, 224)

                    face_img = ImageOps.fit(Image.fromarray(face_img), size, Image.LANCZOS)

                    face_img = (np.asarray(face_img).astype(np.float32) / 127.5) - 1

                    face_img=np.expand_dims(face_img,axis=0)

                    pred=eyemodel.predict(face_img)[0][0]
```

```
          if(pred>0.9):
            path="data/"+str(i)+".jpg"
            cv2.imwrite(path,frame[y:y+h,x:x+w])
            cv2.rectangle(frame,(x,y),(x+h,y+w),(0,0,255),3)
          else:
            cv2.rectangle(frame,(x,y),(x+h,y+w),(0,255,0),3)
        window.image(frame,channels="BGR")
elif(choice=="CAMERA"):
  cam=st.selectbox("Choose 0 for primary camera and 1 for secondary camera",("None",0,1))
  btn=st.button("Start Detection")
  window=st.empty()
  if btn:
    i=1
    btn2=st.button('Stop detection')
    facemodel=cv2.CascadeClassifier("face.xml")
    eyemodel=load_model("eyes.h5",compile=False)
    vid=cv2.VideoCapture(cam)
    if btn2:
      vid.release()
      st.experimental_rerun()
    while(vid.isOpened()):
      flag,frame=vid.read()
      if(flag):
        faces=facemodel.detectMultiScale(frame)
        for(x,y,w,h) in faces:
          face_img=frame[y:y+h,x:x+w]
          size = (224, 224)
```

```
                face_img = ImageOps.fit(Image.fromarray(face_img), size, Image.LANCZOS)

                face_img = (np.asarray(face_img).astype(np.float32) / 127.5) - 1

                face_img=np.expand_dims(face_img,axis=0)

                pred=eyemodel.predict(face_img)[0][0]

                if(pred>0.9):

                   path="data/"+str(i)+".jpg"

                   cv2.imwrite(path,frame[y:y+h,x:x+w])

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

                else:

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

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

elif(choice=="Feedback"):

    st.markdown('<iframe src="https://docs.google.com/forms/d/e/1FAIpQLSfKoxF-E03CCFog8ycsyNiaAkBqRh9Is-HbnhGQbfIl7_aUDw/viewform?embedded=true" width="640" height="1214" frameborder="0" marginheight="0" marginwidth="0">Loading…</iframe>',unsafe_allow_html=True)
```

5. System Architecture:

Backend:

- **Connection with IP Camera:** Utilizes OpenCV for establishing connections with IP cameras.
- **Face Detection:** OpenCV for detecting faces within the captured frames.
- **Eye Detection:** PIL for detecting eyes within the region of interest (ROI).
- **Data Storage:** OpenCV for saving data when closed eyes are detected.

Frontend:

- **User Interface:** Developed using Streamlit to create an interactive and user-friendly web application.

6. Workflow:

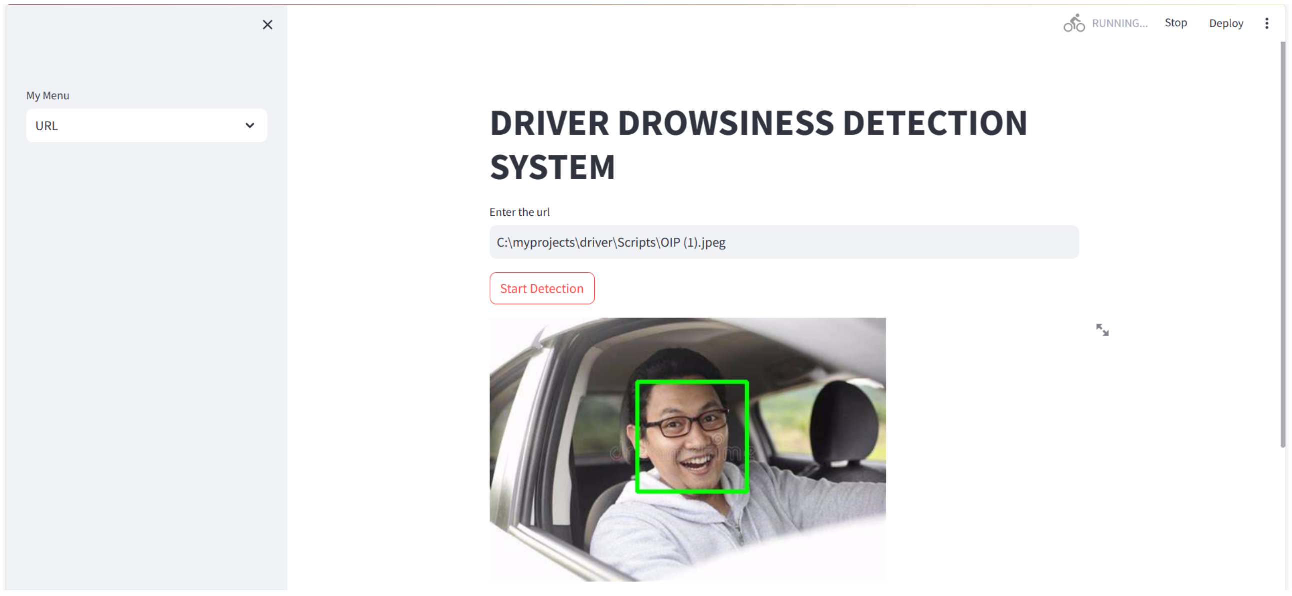
The system captures video frames from the selected source (IP camera, webcam, or online video).

OpenCV performs face detection within the frames.

PIL is employed to detect the state of the eyes within the identified facial region.

If closed eyes are detected, OpenCV stores relevant data for analysis.

Streamlit provides a web-based interface for users to access and interact with the system.



DRIVER DROWSINESS DETECTION SYSTEM

Enter the url

C:\myprojects\driver\Scripts\OIP (1).jpeg

7. Conclusion:

The Driver Drowsiness Detection System is a vital tool in addressing safety concerns related to drowsy driving. By leveraging computer vision techniques and a web-based interface, the system ensures real-time monitoring of drivers and provides data for further analysis. Its applications in the automotive industry, public transportation, and research make it a versatile solution for promoting road safety and driver well-being.

8. Future Enhancements:

Future enhancements may include:

- Integration with advanced machine learning models for more accurate eye state detection.
- Real-time alerts and notifications to drivers and fleet managers.
- Continuous refinement of the user interface for a seamless and intuitive experience.