

# Drowsiness Detection

USING OPENCV AND PYTHON

TEAM

P SUSHMA

M MOHAN

G GAYATRI

CH DEVA RAJU

GUIDE

MR. S.V.R VARA PRASAD



# Today's agenda

Introduction  
Abstract  
Existing Work  
Proposed Work  
Algorithm on Proposed Work  
Architecture

Software& HardWare Requirements  
Data Flow Diagram  
UML Diagrams  
Sample Code  
Output Screens  
References

# Introduction

1. The attention position of driver degrades because of lower sleep, long nonstop driving or any other medical condition like brain diseases etc.
2. Several checks on road accidents says that around 30 percent of accidents are caused by fatigue of the motorist. When driver drives for further than normal period for mortal also inordinate fatigue is caused and also results in frazzle which drives the motorist to sleepy condition or loss of knowledge.
3. Drowsiness is a complex miracle which states that there's a drop in cautions and conscious situations of the driver.

# Abstract

This document is a review report on the research conducted and the project made in the field of computer engineering to develop a system for driver drowsiness detection to prevent accidents from happening because of driver fatigue and sleeping. The report proposed the results and solutions on the limited implementation of the various techniques that are introduced in the project. Whereas the implementation of the project gives the real world idea of how the system works and what changes can be done in order to improve the utility of the overall system. Furthermore, the paper states the overall of the observation made by the authors in order to help further optimization in the mentioned field to achieve the utility at a better efficiency for a safer road.

# Existing Work



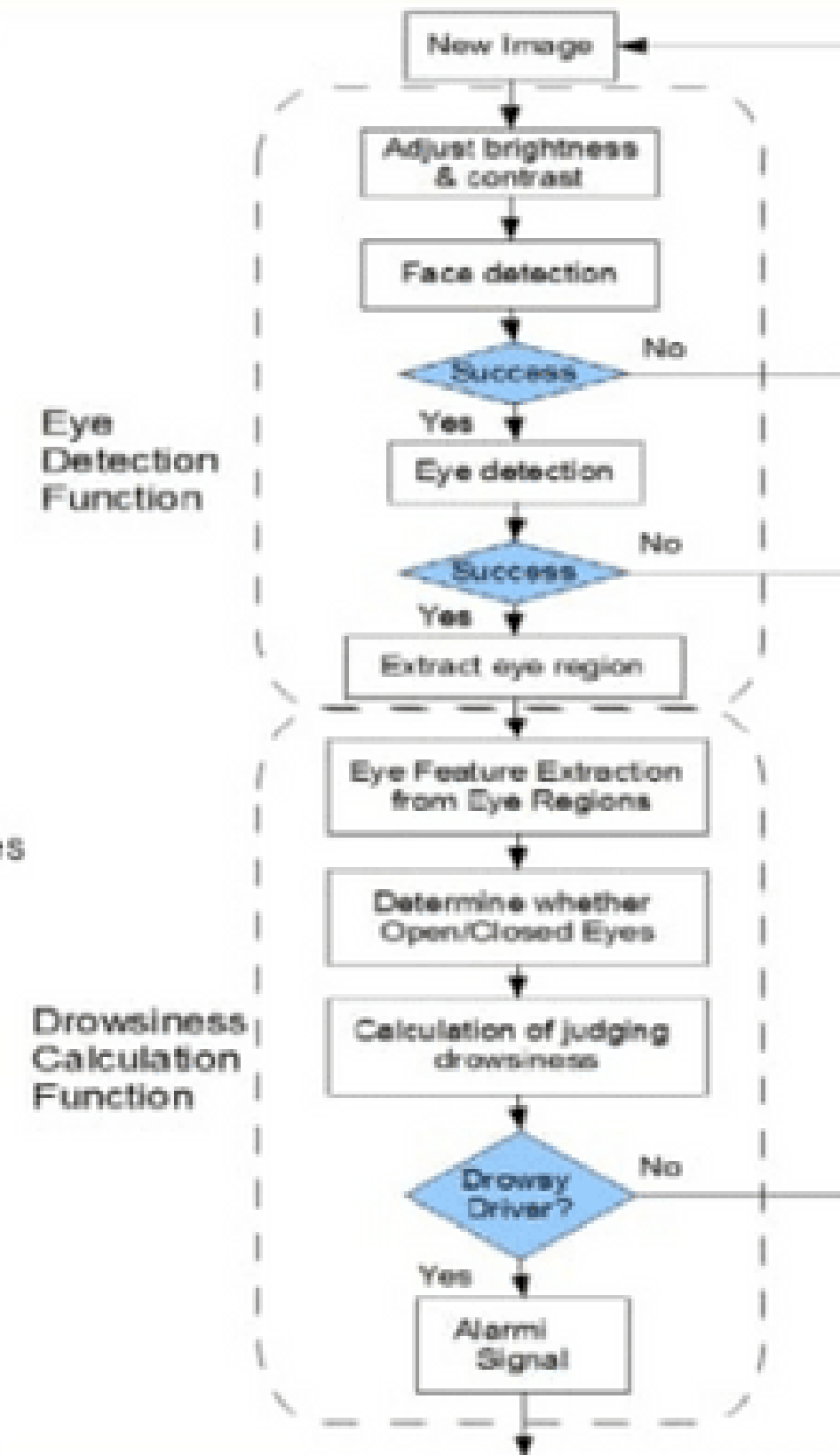
The existing system of driver drowsiness detection system has following **disadvantages**.

Mainly, **using of two cameras** in the system one for monitoring the head movement and the other one for facial expressions. The other disadvantage is aging of sensors and all these sensors are attached to the driver's body which may affect the driver. So to overcome all these disadvantages we designed a system in which a live camera is used for monitoring the driver drowsiness condition and alert the driver which reduces the road accidents.

# Algoritm of Proposing System

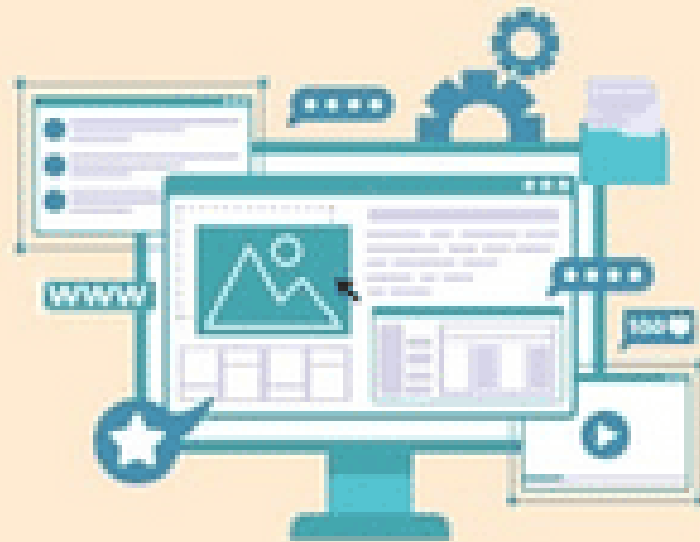
The diagramatic representation of the algorithm shows how the detection of drowsiness happens

- Camera captures the image of driving person.
- Based on the movements of the eyes of a person the algorithm determines whether the person is active or Drowsy.





# Architecture Of our System



1. Face Detection
2. Eye Detection
3. Face Tracking
4. Eye Tracking
5. Drowsiness Detection
6. Alerting

# Software & Hardware Requirements

## Software Requirements Specification


- Python: Python 3.6 and higher version
- Libraries
- Numpy, Scipy , Playsound, Dlib, Imutils, opencv, etc.
- Operating System
- Windows or Ubuntu

## Hardware Requirements Specification

- Processor: 64 bit, quad-core, 2.5 GHz minimum per core
- RAM: 4 GB or more.
- HDD: 20 GB of available space or more.
- Display: Dual XGA (1024 x 768) or higher resolution monitors.
- Camera: A detachable webcam.
- Keyboard: A standard keyboard



---

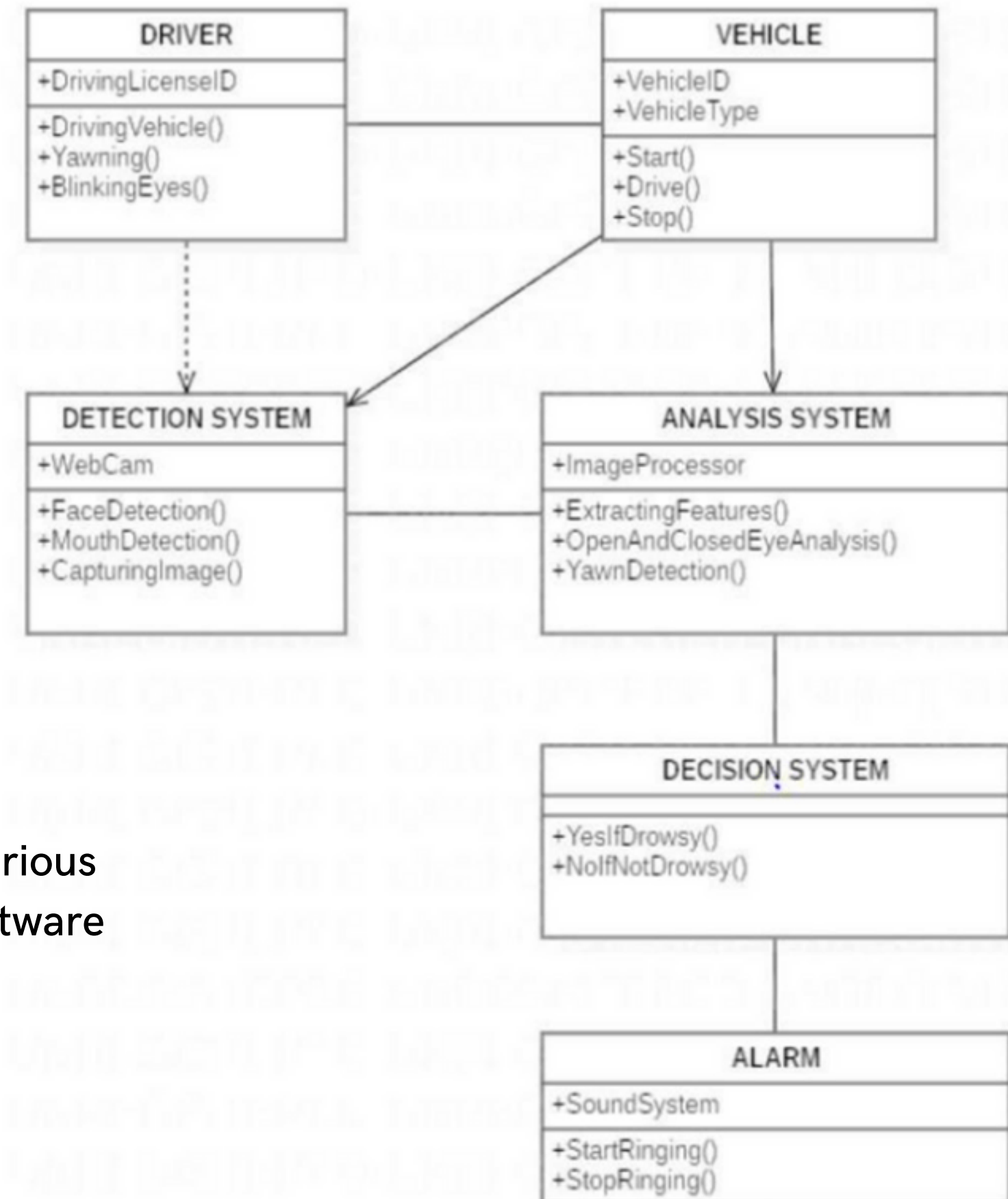


# UML Diagrams

- Class Diagram
- Activity Diagram
- Use Case Diagram
- Sequence Diagram
- State Chart Diagram

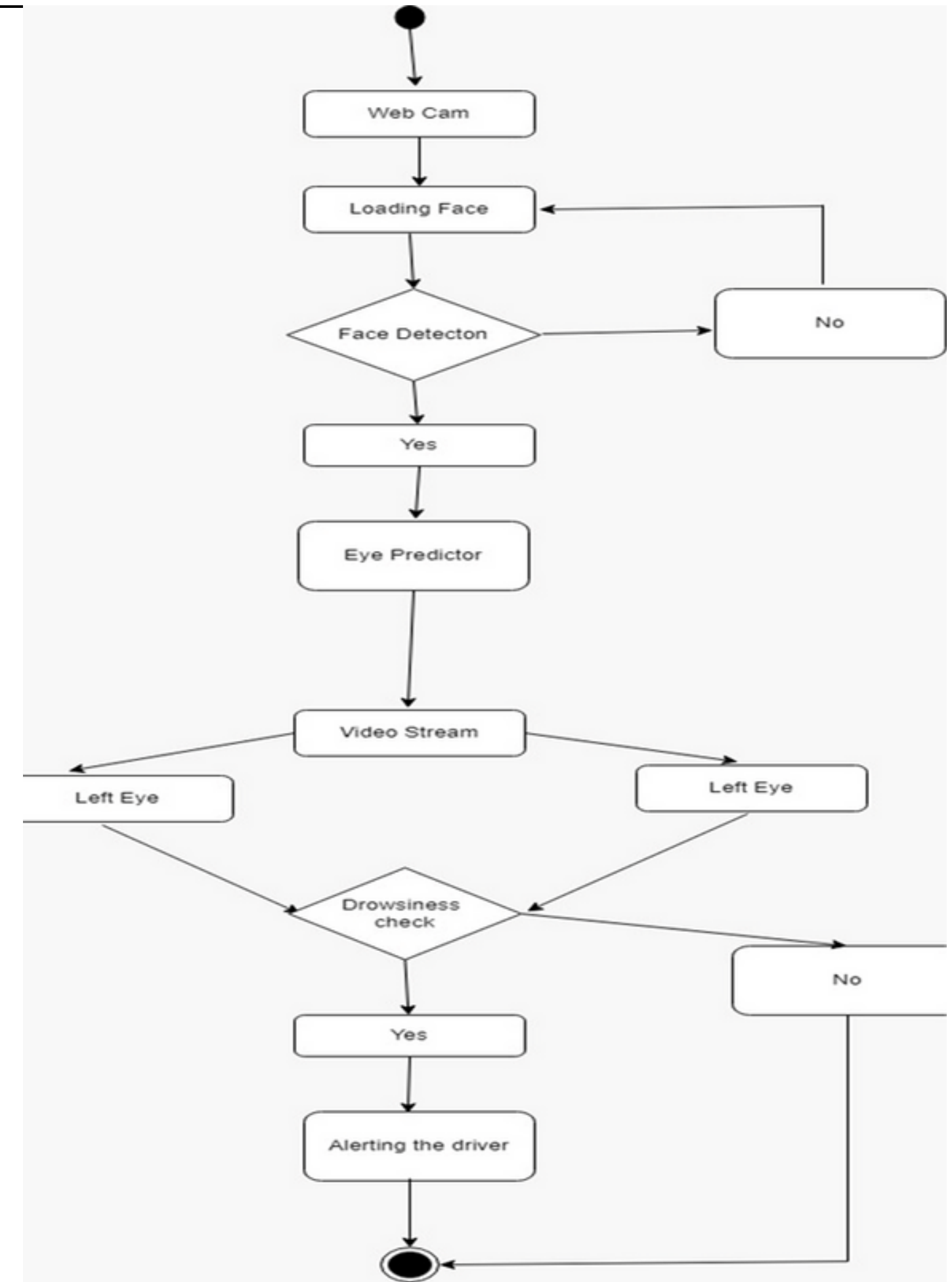
# Class Diagram

- Class diagrams depicts a static view of an application
- You can use class diagrams to visualize, describe, document various different aspects of the system, and also construct executable software code



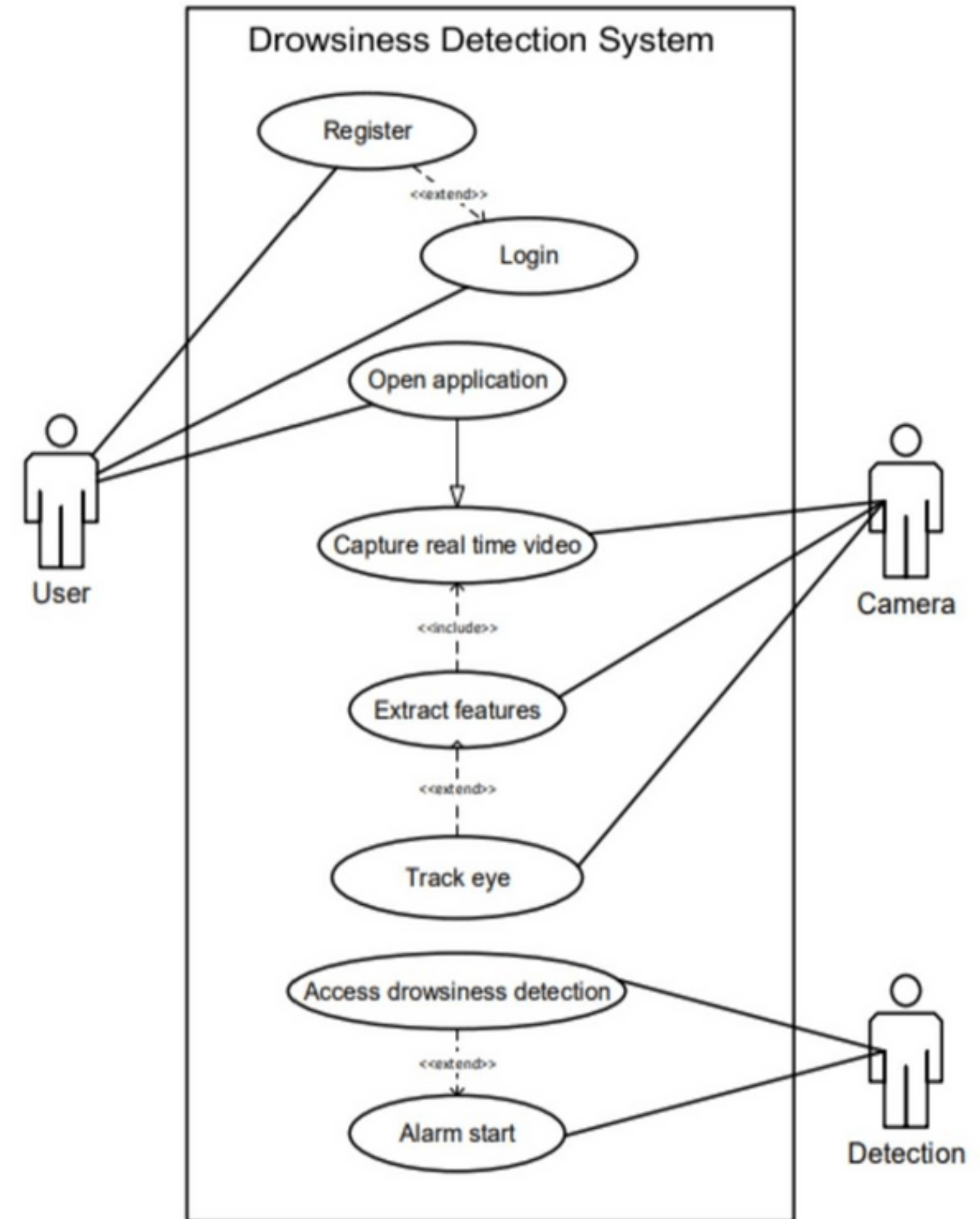
# Activity Diagram

- Activity diagram is basically a flowchart to represent the flow from one activity to another activity.
- The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent.
- Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc



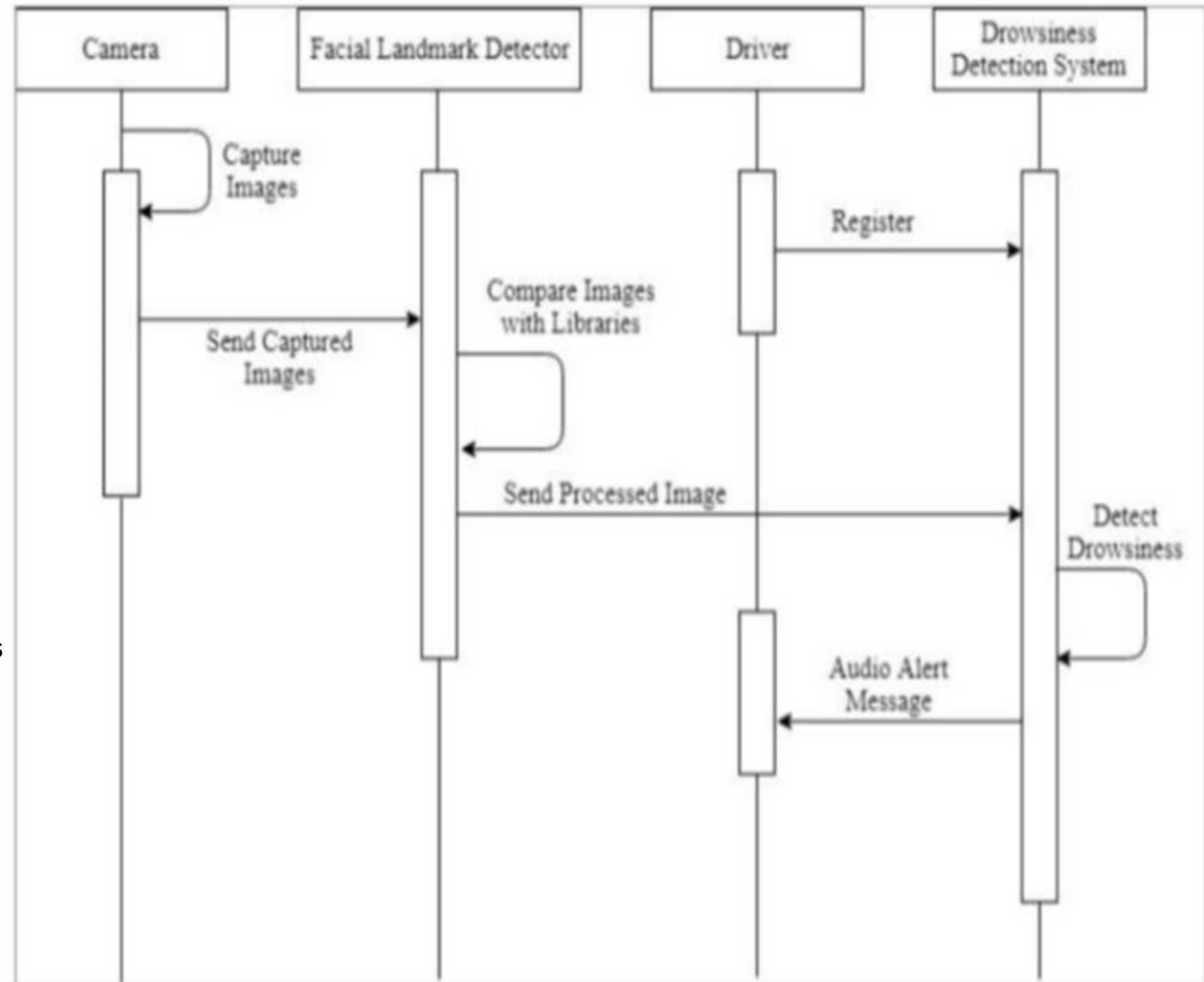
# Use Case Diagram

- Use-case diagrams model the behavior of a system and help to capture the requirements of the system
- It is used to represent the dynamic behavior of the system.



# Sequence Diagram

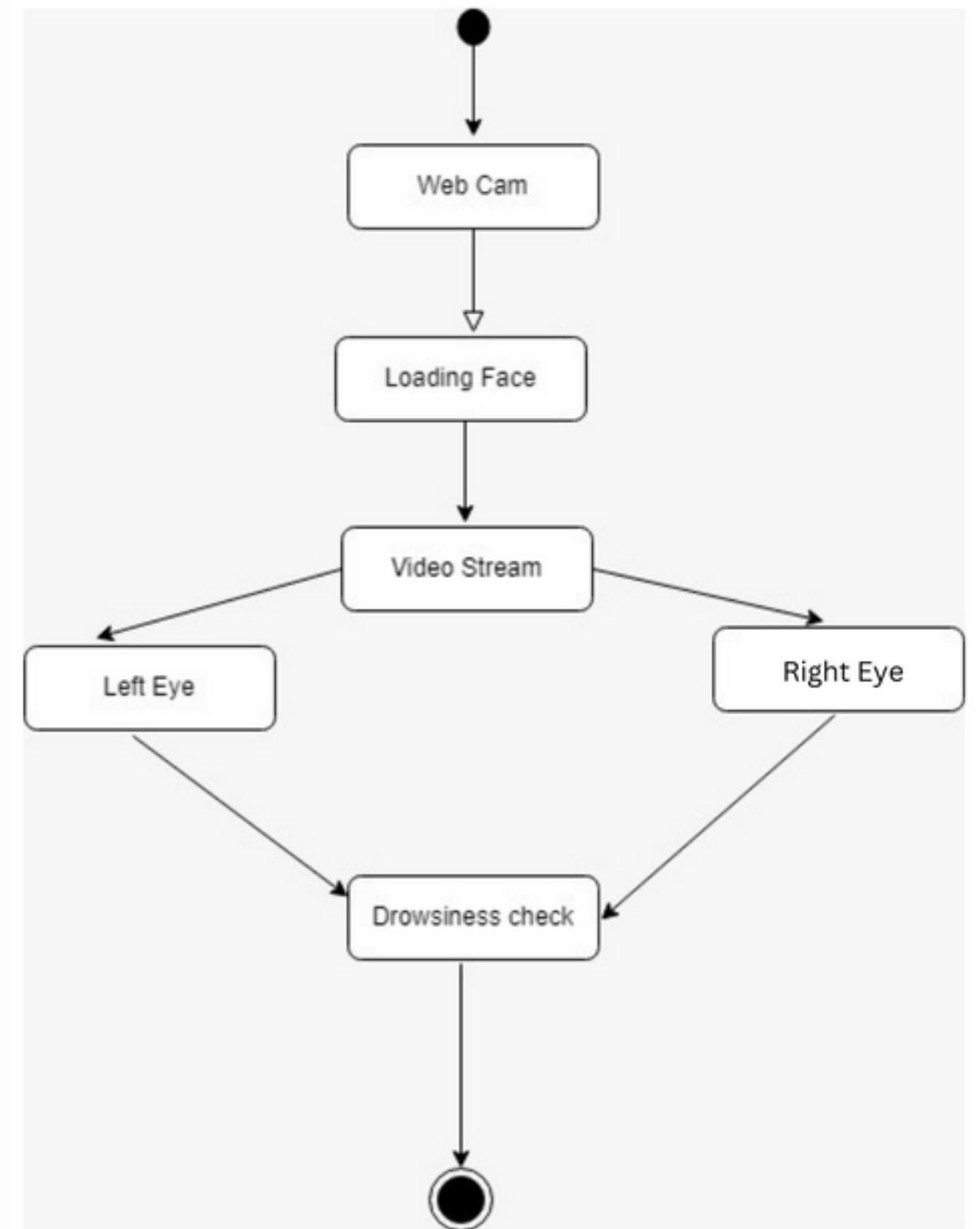
- It represents a timeline that begins at the top and descends gradually to mark the sequence of interactions
- Main purpose is to document and understand requirements for new and existing systems



# State Chart Diagram

A state diagram, also known as a state machine diagram or statechart diagram, is an illustration of the states an object can attain as well as the transitions between those states in the Unified Modeling Language (UML).

State diagrams are used to give an abstract description of the behavior of a system. This behavior is analyzed and represented by a series of events that can occur in one or more possible states.



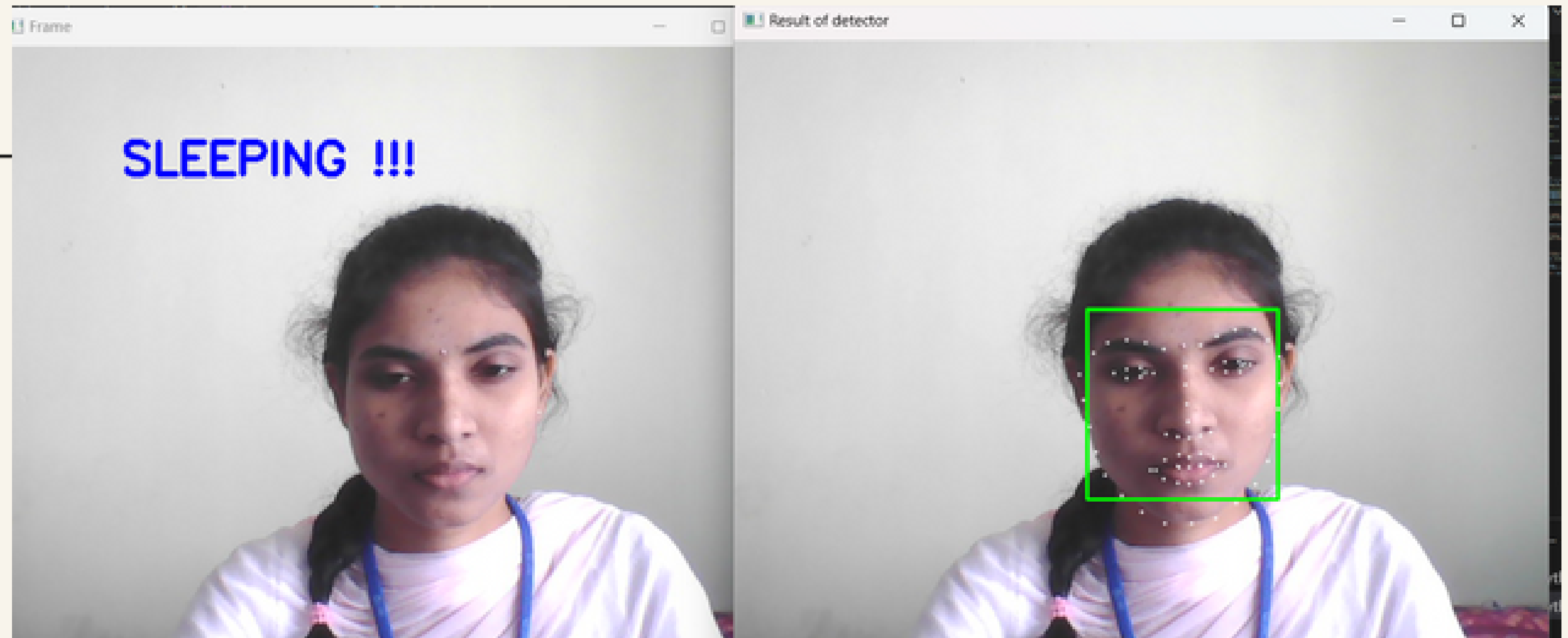


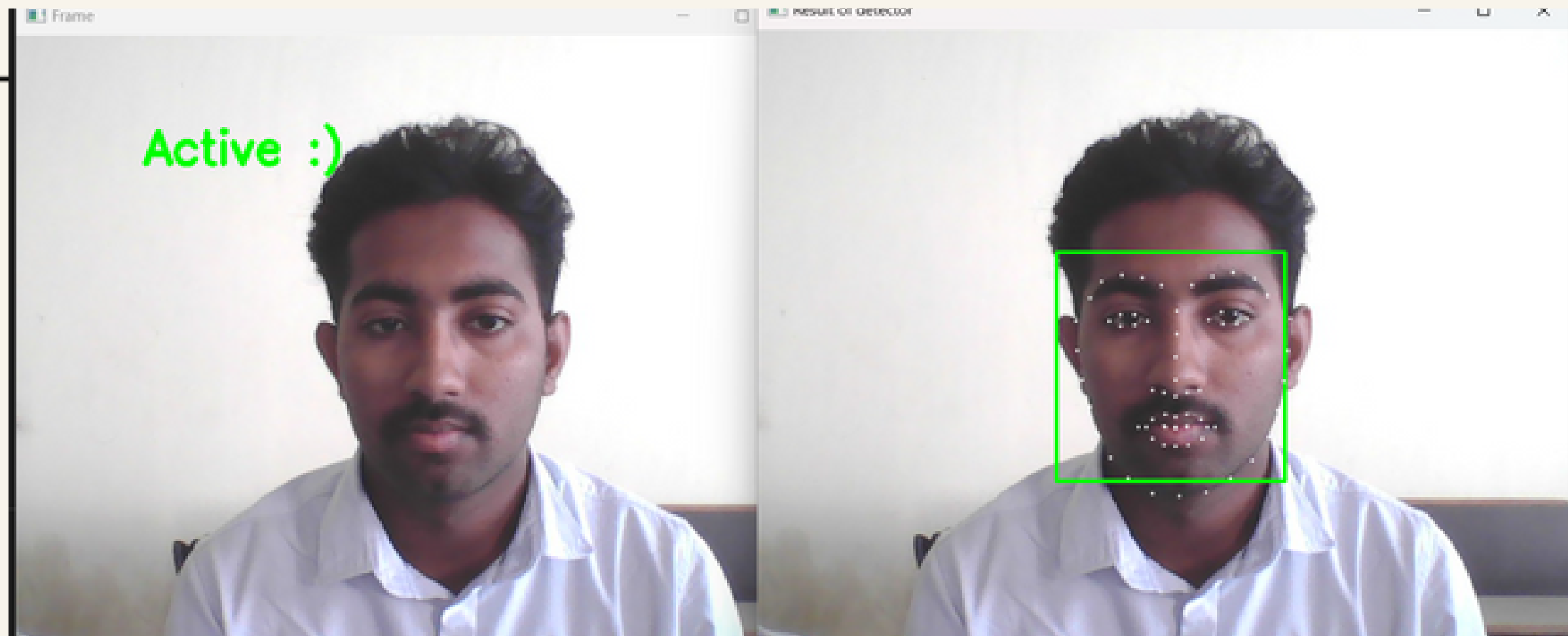
# Sample Code:



```
import cv2
# Numpy for array related functions
import numpy as np
# Dlib for deep learning based Modules and face landmark detection
import dlib
#face_utils for basic operations of conversion
from imutils import face_utils
#Initializing the camera and taking the instance
cap = cv2.VideoCapture(0)
#Initializing the face detector and landmark detector
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
#status marking for current state
sleep = 0
drowsy = 0
active = 0
status=""
color=(0,0,0)
def compute(ptA,ptB):
    dist = np.linalg.norm(ptA - ptB)
    return dist
def blinked(a,b,c,d,e,f):
    up = compute(b,d) + compute(c,e)
    down = compute(a,f)
    ratio = up/(2.0*down)
    #Checking if it is blinked
    if(ratio>0.25):
        return 2
    elif(ratio>0.21 and ratio<=0.25):
        return 1
    else:
        return 0
```

# OUTPUT SCREENS





# Conclusion:

- An automated technique for identifying driver tiredness was created in the current study.
- The continuous video stream is read from the system and is used for detecting the drowsiness.
- Using the Haar cascade method, it is discovered. Haar characteristics are used by the haar cascade algorithm to find faces and eyes.



# References

1. A. Malla, P. Davidson, P. Bones, R. Green and R. Jones, "Automated Video-based Measurement of Eye Closure for Detecting Behavioral Microsleep", in 32nd Annual International Conference of the IEEE, Buenos Aires, Argentina, 2010.
2. P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001.
3. R. Lienhart and J. Maydt, "An Extended Set of Haar-like Features for Rapid Object Detection", in Proceedings of the IEEE International Conference on Image Processing, 2002.
4. S. Vitabile, A. Paola and F. Sorbello, "Bright Pupil Detection in an Embedded, Real-time Drowsiness Monitoring System", in 24th IEEE International Conference on Advanced Information Networking and Applications, 2010.
5. B. Bhowmick and C. Kumar, "Detection and Classification of Eye State in IR Camera for Driver Drowsiness Identification", in Proceeding of the IEEE International Conference on Signal and Image Processing Applications, 2009.

