**Chapter 1**

**Introduction**

**1.1 Project:**

Real Time Drowsiness behaviors which are related to fatigue are in the form of eye closing, head nodding or the brain activity. Hence, we can either measure change in physiological signals, such as brain waves, heart rate and eye blinking to monitor drowsiness or consider physical changes such as sagging posture, leaning of driver’s head and open/closed state of eyes.

The former technique, while more accurate, is not realistic since highly sensitive electrodes would have to be attached directly on the driver’ body and hence which can be annoying and distracting to the driver. In addition, long time working would result in perspiration on the sensors, diminishing their ability to monitor accurately. The second technique is to measure physical changes (i.e. open/closed eyes to detect fatigue) is well suited for real world conditions since it is non-intrusive by using a video camera to detect changes. In addition, micro sleeps that are short period of sleeps lasting 2 to 3 minutes are good indicators of fatigue. Thus, by continuously monitoring the eyes of the driver one can detect the sleepy state of driver and a timely warning is issued.

**1.2 Machine learning:**

Machine learning (ML) is a category of algorithm that allows software applications to become more accurate in predicting outcomes without being explicitly programmed. The basic premise of machine learning is to build algorithms that can receive input data and use statistical analysis to predict an output while updating outputs as new data becomes available. Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed .Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

**Supervised learning:**

Supervised learning as the name indicates the presence of a supervisor as a teacher. Basically supervised learning is a learning in which we teach or train the machine using data which is well labeled that means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples(data) so that supervised learning algorithm analyses the training data(set of training examples) and produces a correct outcome from labeled data.Machine learning algorithms are often categorized as supervised or unsupervised. Supervised algorithms require a data scientists or data analyst with machine l[u](https://whatis.techtarget.com/definition/unsupervised-learning)earning skills to provide both input and desired output, in addition to furnishing feedback about the accuracy of predictions during algorithm training. Data scientists determine which variables, or features, the model should analyze and use to develop predictions. Once training is complete, the algorithm will apply what was learned to new data.

**Unsupervised Learning:** Unsupervised algorithms do not need to be trained with desired outcome data. Instead, they use an iterative approach called deep learning to review data and arrive at conclusions. Unsupervised learning algorithms -- also called neural network -- are used for more complex processing tasks than supervised learning systems, including image recognition, speech-to-text and natural language generation. These neural networks work by combing through millions of examples of training data and automatically identifying often subtle correlations between many variables. Once trained, the algorithm can use its bank of associations to interpret new data. These algorithms have only become feasible in the age of big data, as they require massive amounts of training data.

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that which makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

**Chapter 2**

**Review of Literature**

In this section, we have discussed various methodologies that have been proposed by researchers for drowsiness detection and blink detection during the recent years.

Manu B.N in 2016, has proposed a method that detect the face using Haar feature-based cascade classifiers. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier that will detect the object. So along with the Haar feature-based classifiers, cascaded Adaboost classifier is exploited to recognize the face region then the compensated image is segmented into numbers of rectangle areas, at any position and scale within the original image. Due to the difference of facial feature, Haar-like feature is efficient for real-time face detection. These can be calculated according to the difference of sum of pixel values within rectangle area and during the process the Adaboost algorithm will allow all the face samples and it will discard the non-face samples of images.

Amna Rahman in 2015, has proposed a method to detect the drowsiness by using Eye state detection with Eye blinking strategy. In this method first, the image is converted to gray scale and the corners are detected using Harris corner detection algorithm which will detect the corner at both side and at down curve of eye lid. After tracing the points then it will make a straight line between the upper two points and locates the mid-point by calculation of the line, and it connects the mid-point with the lower point. Now for each image it will perform the same procedure and it calculates the distance ‘d’ from the mid-point to the lower point to determine the eye state. Finally, the decision for the eye state is made based on distance ’d’ calculated. If the distance is zero or is close to zero, the eye state is classified as “closed” otherwise the eye state is identified as “open”. They have also invoked intervals or time to know that the person is feeling drowsy or not. This is done by the average blink duration of a person is 100-400 milliseconds (i.e. 0.1-0.4 of a second).

**Chapter 3**

**Software and Hardware Requirements**

**Software Requirements**

Operating System : Windows7/8/10

Front End : Python

IDE : Spyder

**Hardware Requirements**

Processor : Pentium IV processor

Hard Disk : 25 GB

RAM : 256 MB

**Chapter 4**

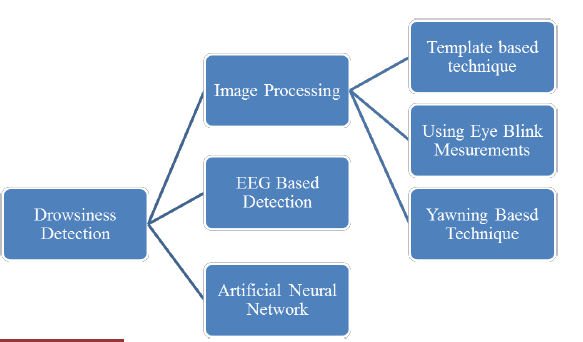
**Proposed Methodology**

**3.1 The different types of methodologies have been developed to find out drowsiness.**

**3.1.1 Physiological level approach:** This technique is an intrusive method whereinelectrodes are used to obtain pulse rate, heart rate and brain activity information. ECG is used to calculate the variations in heart rate and detect different conditions for drowsiness. The correlation between different signals such as ecg (electrocardiogram), EEG (electroencephalogram), and EMG (electromyogram) are made and then the output is generated whether the person is drowsy or not.

**3.1.2 Behavioral based approach:** In this technique eye blinking frequency, head pose, etc.of a person is monitored through a camera and the person is alerted if any of these drowsiness symptoms are detected.

**3.2 Various Drowsiness detection techniques:**

****

***Fig 1***

1. **Images Processing Based Techniques**

In image processing based techniques, drivers face images are used for processing so that one can find its states. From the face image one can see that driver is awake or sleeping. Using same images, they can define drowsiness of driver because in face image if driver is sleeping or dozing then his/her eyes are closed in image. And other symptoms of drowsiness can also detected from the face image. We can classify these techniques in three sub-categories.

**a) Template Matching Technique:**

In this technique, one can use the states of eye i.e. if driver closes eye/s for some particular time then system will generate the alarm.Becasue in this techniques system has both close and open eyes template of driver. This system can also be trained to get open and closed eye templates of driver.

Open and closed eyes template.This method is simple and easy to implement because templates of both open and closed eye states shown in figure are available to system. Researchers have used this technique.

**b) Eye Blinking based Technique:**

In this eye blinking rate and eye closure duration is measured to detect driver’s drowsiness. Because when driver felt sleepy at that time his/her eye blinking and gaze between eyelids are different from normal situations so they easily detect drowsiness. In this system the position of irises and eye states are monitored through time to estimate eye blinking frequency and eye close duration And in this type of system uses a remotely placed camera to aquire video and computer vision method esare then applied to sequentially localize face, eyes and eyelids positions to measure ratio of closure. Using these eyes clouser and blinking ration one can detect drowsiness of driver.

**c) Yawning Based Technique***:*

Yawn is one of the symptoms of fatigue. The yawn is assumed to be modeled with a large vertical mouth opening. Mouth is wide open is larger in yawning compared to speaking. Using face tracking and then mouth tracking one can detect yawn. they detect yawning based on opening rate of mouth and the amount changes in mouth contour area. When yawn is detected by system then it alarm the driver. Instead of using just one technique to detect drowsiness of driver, some researchers [1, 2, 3] have combined various vision based image processing techniques to have better performances.

**2) EEG Based Technique**

In this technique it is compulsory to wear electrode helmet by drivers while driving . This helmet have various electrode sensors which placed at correct place and get data from brain. Researchers have used the characteristic of EEG signal in drowsy driving. A method based on power spectrum analysis and Fast ICA algorithm was proposed to determining the fatigue degree. In a driving simulation system, the EEG signals of subjects were captured by instrument NT-9200 in two states, one state was sober, and the other was drowsy. The multi- channel signals were analyzed with Fast ICA algorithm, to remove ocular electric, my electric and power frequency interferences. Figure 4 shows how EEG based systems get data for acquisition. Experimental results show that the method presented in this paper can be used to determine the drowsiness degree of EEG signal effectually

**3.2 The various technology that can be used are discussed as:**

**3.2.1 Tensor Flow:** IT is an [open-source](https://en.m.wikipedia.org/wiki/Open-source) [software library](https://en.m.wikipedia.org/wiki/Library_%28computing%29) for [dataflow programming](https://en.m.wikipedia.org/wiki/Dataflow_programming) across a range of tasks. It is a symbolic math library, and is also used for [machine learning](https://en.m.wikipedia.org/wiki/Machine_learning) applications such as [neural networks.](https://en.m.wikipedia.org/wiki/Neural_networks) It is used for both research and production.

Tensor Flow computations are expressed as stateful dataflow [graphs.](https://en.m.wikipedia.org/wiki/Directed_graph) The name Tensor Flow derives from the operations that such neural networks perform on multidimensional data arrays. These arrays are referred to as ["tensors".](https://en.m.wikipedia.org/wiki/Tensor)

**3.2.2 Machine learning**: Machine learning is the kind of programming which givescomputers the capability to automatically learn from data without being explicitly programmed. This means in other words that these programs change their behavior by learning from data. Python is clearly one of the best languages for machine learning. Python does contain special libraries for machine learning namely scipy, pandas and numpy which great for linear algebra and getting to know kernel methods of machine learning. The language is great to use when working with machine learning algorithms and has easy syntax relatively.

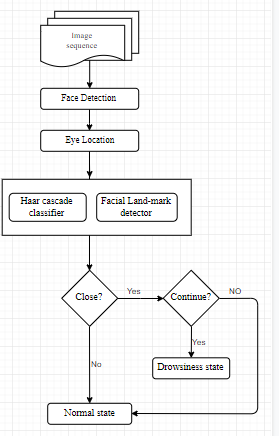
**3.2.3 OpenCV:** OpenCV stands for Open Source Computer Vision. It's an Open Source BSDlicensed library that includes hundreds of advanced Computer Vision algorithms that are optimized to use hardware acceleration. OpenCV is commonly used for machine learning, image processing, image manipulation, and much more. OpenCV has a modular structure.

There are shared and static libraries and a CV Namespace.

In short, OpenCV is used in our application to easily load bitmap files that contain landscaping pictures and perform a blend operation between two pictures so that one picture can be seen in the background of another picture. This image manipulation is easily performed in a few lines of code using OpenCV versus other methods. OpenCV.org is a must if you want to explore and dive deeper into image processing and machine learning in general.

**3.2.4) Kivy:** Kivy is an open source [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) library for developing [mobile apps](https://en.wikipedia.org/wiki/Mobile_app) and other [multi touch](https://en.wikipedia.org/wiki/Multitouch) [application software](https://en.wikipedia.org/wiki/Application_software) with a [natural user interface (NUI).](https://en.wikipedia.org/wiki/Natural_User_Interface) It can run on [Android,](https://en.wikipedia.org/wiki/Android_(operating_system)) [iOS,](https://en.wikipedia.org/wiki/IOS) [Linux,](https://en.wikipedia.org/wiki/Linux) [OS X,](https://en.wikipedia.org/wiki/OS_X) and [Windows.](https://en.wikipedia.org/wiki/Microsoft_Windows) Distributed under the terms of the [MIT license,](https://en.wikipedia.org/wiki/MIT_License) Kivy is [free](https://en.wikipedia.org/wiki/Free_Software) and [open source software.](https://en.wikipedia.org/wiki/Open_source_software) Kivy is the main framework developed by the Kivy organization, alongside Python for Android, Kivy iOS, and several other libraries meant to be used on all platforms.

**3.4 System Description**

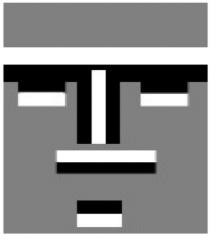
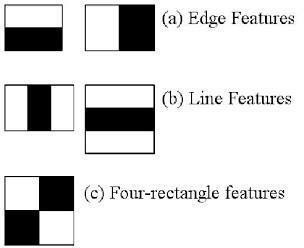
****

***Fig 1***

The various detection stages are discussed as:

**1. Face Detection:** For the face Detection it uses Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted cascade of simple features” in 2001. it is a machine learning based approach where a cascade function is trained from a lot of of positive and negative images . It is then used to detect objects in other images.

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features shown in the below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.

*** Fig 2 Fig 3***

* A cascaded Adaboost classifier with the Haar-like features is exploited to find out the face region. First, the compensated image is segmented into numbers of rectangle areas, at any position and scale within the original image. Due to the difference of facial feature, Haar-like feature is efficient for real-time face detection. These can be calculated according to the difference of sum of sum of pixel values within rectangle areas. The features can be represented by the different composition of the black region and white region .
* These can be calculated according to the difference of the sum of pixel values within rectangle area. The features can be represented by the different composition of the black and white region. A cascaded strong classifier which is a combination of several weak classifiers each weak classifier is trained by Adaboost algorithm. If a candidate sample passes through the cascaded adaboost classifier, the face region can be found. Almost all of our samples can pass through and non-face samples can be rejected.

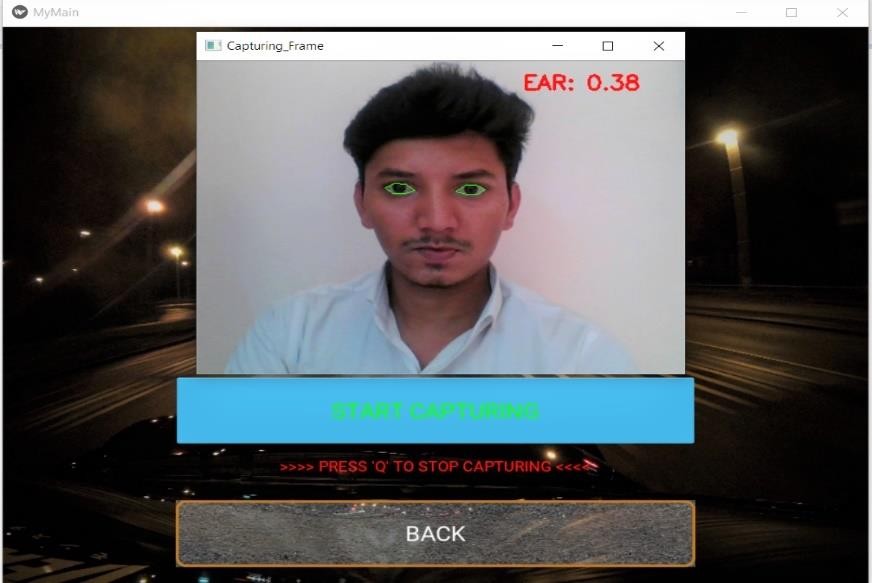
**2. Eye detection:** In the system we have used facial landmark prediction for eye detection facial landmarks are used to localize and represent salient regions of the face , such as:

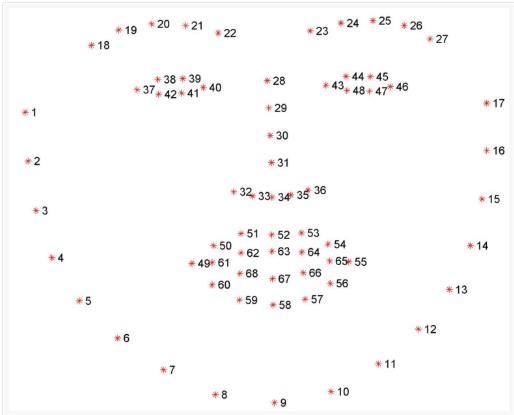
* + - * Eyes
      * Eyebrows
      * Nose
      * Mouth
      * Jawline
* Facial landmarks have been successfully applied to face alignment, head pose estimation, face swapping, blink detection and much more. In the context of facial landmarks, our goal is detecting important facial structures on the face using shape prediction methods. Detecting facial landmarks is therefore a two step process:
  + - * Localize the face in the image.
      * Detect the key facial structures on the face ROI.
* Localize the face in the image: The face image is localized by Haar feature-based cascade classifier which was discussed in the first step of our algorithm i.e. face detection.
* Detect the key facial structures on the face : there are variety of facial landmark detectors , but all methods essentially try to localize and label the following facial region
  + - * Mouth
      * Righteyebrow
      * Lefteyebrow
      * Righteye
      * Lefteye
      * Nose

The facial landmark detector included in the dlib library is an implementation of the[OneMillisecond Face Alignment with an Ensemble of Regression Trees](https://pdfs.semanticscholar.org/d78b/6a5b0dcaa81b1faea5fb0000045a62513567.pdf)paper by Kazemi and Sullivan (2014).

This method starts by using:

* A training set of labeled facial landmarks on an image. These images are manually labeled, specifying specific (x, y*)*-coordinates of regions surrounding each facial structure.
* Priors, of more specifically, the probability on distance between pairs of input pixels. The pre-trained facial landmark detector inside the dlib library is used to estimate the location of 68 (x, y)-coordinates that map to facial structures on theface.
* The indexes of the 68 coordinates can be visualized on the imagebelow:

***Fig 4:*** *Detection of both the eyes*



***Fig 5:*** *Visualizing the 68 facial landmark coordinates*

* We can detect and access both the eye region by the following facial landmark index show below
  + The right eye using [36,42].
  + The left eye with [42,48].
* These annotations are part of the 68 point[iBUG 300-W dataset](https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/)which the dlib facial landmark predictor was trained on. It’s important to note that other flavors of facial landmark detectors exist, including the 194 point model that can be trained on the[HELEN dataset](http://www.ifp.illinois.edu/~vuongle2/helen/).
* Regardless of which dataset is used, the same dlib framework can be leveraged to train a shape predictor on the input training data.

**3. Recognition of Eye's State:**

* The eye area can be estimated from optical flow, by sparse tracking or by frame-to-frame intensity differencing and adaptive thresholding. And finally, a decision is made whether the eyes are or are not covered by eyelids. A different approach is to infer the state of the eye opening from a single image, as e.g. by correlation matching with open and closed eye templates, a heuristic horizontal or vertical image intensity projection over the eye region, a parametric model fitting to find the eyelids, or active shape models. A major drawback of the previous approach is that they usually implicitly impose too strong requirement on the setup, in the sense of a relative face-camera pose (head orientation), image resolution, illumination, motion dynamics, etc. Especially the heuristic methods that use raw image intensity are likely to be very sensitive despite their real-time performance.
* Therefore, we propose a simple but efficient algorithm to detect eye blinks by using a recent facial landmark detector. A single scalar quantity that reflects a level of the eye opening is derived from the landmarks. Finally, having a per-frame sequence of the eye-opening estimates, the eye blinks are found by an SVM classifier that is trained on examples of blinking and non- blinking patterns.

###### **4. Eye Aspect Ratio Calculation:**

* For every video frame, the eye landmarks are detected. The eye aspect ratio (EAR) between height and width of the eye is computed.
* EAR =||p2 − p6|| + ||p3− p5|| (1)2||p1 −p4||
* wherep1,...,p6 are the 2D landmark locations , depicted .The EAR is mostly constant when an eye is open and is getting close to zero while crossing an eye its partially person and head posein sensitive. Aspect ratio of the open eye has a small variance among individuals, and it is fully invariant to a uniform scaling of the image and in-plane rotation of the face. Since eye blinking is performed by both eyes synchronously, the EAR of both eyes is averaged.

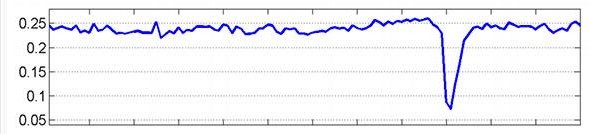
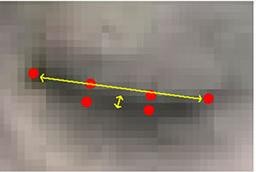
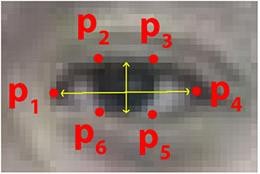
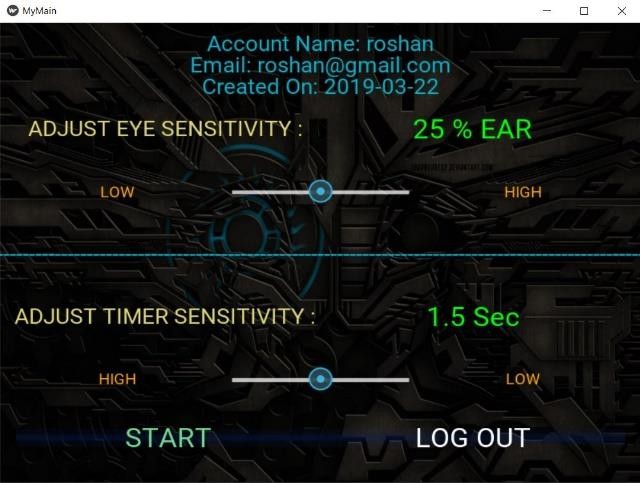


Fig.3.4:Openandclosedeyeswithlandmarks Fig.3.5 EAR for singleblink

* User flexibility Set up Features: This interface helps the user to adjust the EAR accordingly ranging from low to high percent. It also allows the user to set the timer sensitivity shown in the fig 3.6

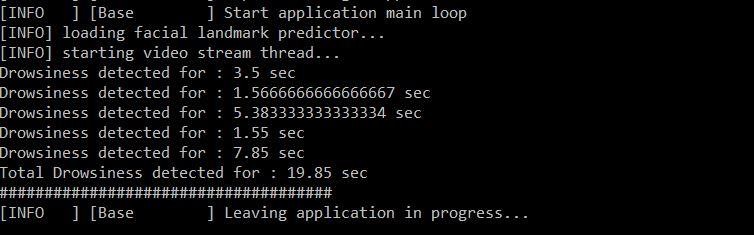
***Fig. 6***

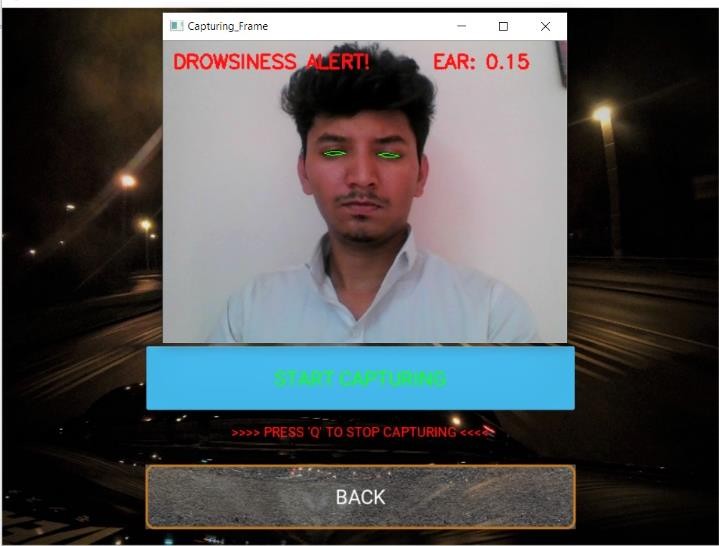
###### **5. Eye State Determination:**

* Finally, the decision for the eye state is made based on EAR calculated in the previous step. If the distance is zero or is close to zero, the eye state is classified as “closed” otherwise the eye state is identified as “open”.

###### **6. Drowsiness Detection:**

* The last step of the algorithm is to determine the person’s condition based on a pre-set condition for drowsiness. The average blink duration of a person is 100-400 milliseconds (i.e. 0.1-0.4 of a second). Hence if a person is drowsy his eye closure must be beyond this interval. We set a time frame of 5 seconds. If the eyes remain closed for five or more seconds, drowsiness is detected and alert pop regarding this is triggered.

***Fig 7 Console information***



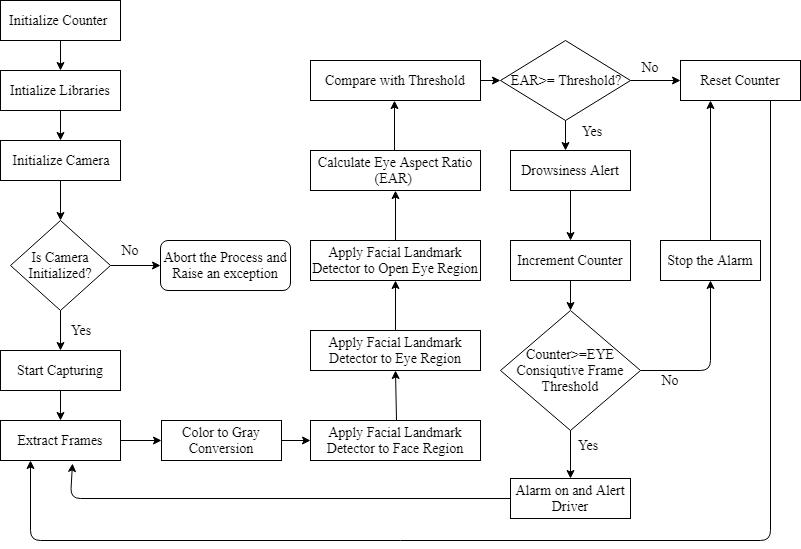
***Fig 8 Drowsiness state***

**7. Working Principle:**

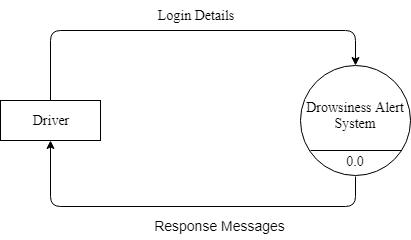
A Drowsy Driver Detection System has been developed, using a non-intrusive machine vision based concepts. The system uses a web camera that points directly towards the driver’s face and monitors the driver’s head movements in order to detect fatigue. In such a case when fatigue is detected, a warning signal is issued to alert the driver. The algorithm developed is unique to any currently published papers, which was a primary objective of the project. The system deals with detecting eyes, nose and mouth within the specific segment of the image.

**Chapter 5**

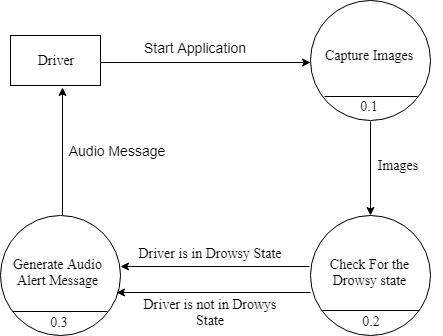
**Modeling diagram (project Work Flow)**



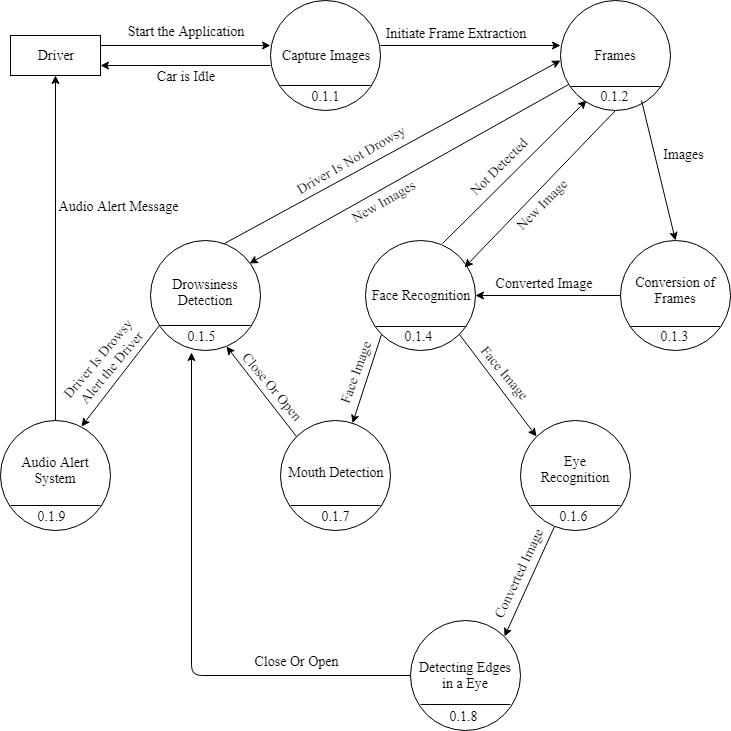
***Fig 9 Flow chart of System***

******

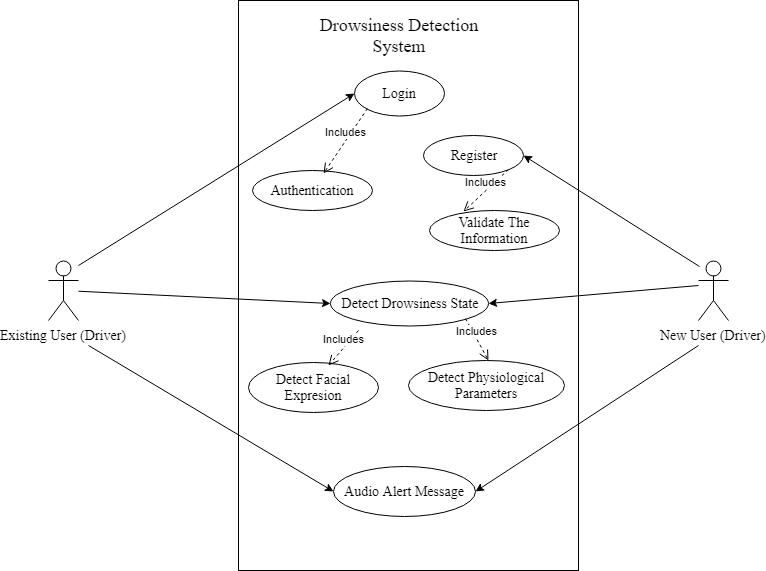
***Fig 10 DFD level 0***



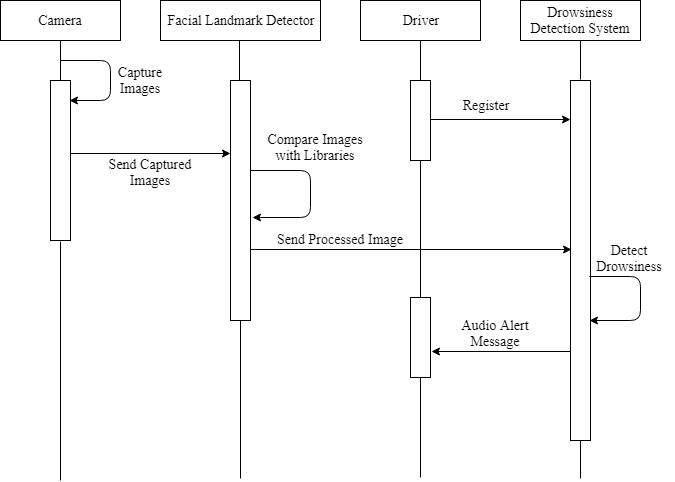
***Fig 11 DFD level 1***

******

***Fig 12 DFD level 2***

******

***Fig 13 use-case Diagram***

******

***Fig 14*** Sequence Diagram

**Chapter 6**

**Coding**

**Modules Used:**

**from \_\_future\_\_ import division**

**import dlib**

**from imutils import face\_utils**

**import cv2**

**import numpy as np**

**from scipy.spatial import distance as dist**

**import threading**

**import sys**

**#import pygame**

**def start\_sound():**

**pygame.mixer.init()**

**pygame.mixer.music.load("z.ogg")**

**pygame.mixer.music.play()**

**def resize(img, width=None, height=None, interpolation=cv2.INTER\_AREA):**

**global ratio**

**w, h = img.shape**

**if width is None and height is None:**

**return img**

**elif width is None:**

**ratio = height / h**

**width = int(w \* ratio)**

**resized = cv2.resize(img, (height, width), interpolation)**

**return resized**

**else:**

**ratio = width / w**

**height = int(h \* ratio)**

**resized = cv2.resize(img, (height, width), interpolation)**

**return resized**

**######**

**def shape\_to\_np(shape, dtype="int"):**

**coords = np.zeros((68, 2), dtype=dtype)**

**for i in range(36,48):**

**coords[i] = (shape.part(i).x, shape.part(i).y)**

**return coords**

**def eye\_aspect\_ratio(eye):**

**A = dist.euclidean(eye[1], eye[5])**

**B = dist.euclidean(eye[2], eye[4])**

**# compute the euclidean distance between the horizontal**

**# eye landmark (x, y)-coordinates**

**C = dist.euclidean(eye[0], eye[3])**

**# compute the eye aspect ratio**

**ear = (A + B) / (2.0 \* C)**

**# return the eye aspect ratio**

**return ear**

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

**predictor\_path = 'shape\_predictor\_68\_face\_landmarks.dat'**

**detector = dlib.get\_frontal\_face\_detector()**

**predictor = dlib.shape\_predictor(predictor\_path)**

**(lStart, lEnd) = face\_utils.FACIAL\_LANDMARKS\_IDXS["left\_eye"]**

**(rStart, rEnd) = face\_utils.FACIAL\_LANDMARKS\_IDXS["right\_eye"]**

**total=0**

**alarm=False**

**while True:**

**ret, frame = camera.read()**

**if ret == False:**

**print('Failed to capture frame from camera. Check camera index in cv2.VideoCapture(0) \n')**

**break**

**frame\_grey = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)**

**frame\_resized = resize(frame\_grey, width=120)**

**# Ask the detector to find the bounding boxes of each face. The 1 in the**

**# second argument indicates that we should upsample the image 1 time. This**

**# will make everything bigger and allow us to detect more faces.**

**dets = detector(frame\_resized, 1)**

**if len(dets) > 0:**

**for k, d in enumerate(dets):**

**shape = predictor(frame\_resized, d)**

**shape = shape\_to\_np(shape)**

**leftEye= shape[lStart:lEnd]**

**rightEye= shape[rStart:rEnd]**

**leftEAR= eye\_aspect\_ratio(leftEye)**

**rightEAR = eye\_aspect\_ratio(rightEye)**

**ear = (leftEAR + rightEAR) / 2.0**

**leftEyeHull = cv2.convexHull(leftEye)**

**rightEyeHull = cv2.convexHull(rightEye)**

**cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)**

**cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)**

**if ear>.25:**

**print (ear)**

**total=0**

**alarm=False**

**cv2.putText(frame,"EyesOpen",(10, 30),cv2.FONT\_HERSHEY\_SIMPLEX, 0.7, (0, 0, 255), 2)**

**else:**

**total+=1**

**if total>20:**

**if not alarm:**

**alarm=True**

**d=threading.Thread(target=start\_sound)**

**d.setDaemon(True)**

**d.start()**

**print ("so jaaaaaaaaaa")**

**cv2.putText(frame, "drowsiness detect" ,(250, 30),cv2.FONT\_HERSHEY\_SIMPLEX, 1.7, (0, 0, 0), 4)**

**cv2.putText(frame, "Eyes close".format(total), (10, 30),cv2.FONT\_HERSHEY\_SIMPLEX, 0.7, (0, 0, 255), 2)**

**for (x, y) in shape:**

**cv2.circle(frame, (int(x/ratio), int(y/ratio)), 3, (255, 255, 255), -1)**

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

**if cv2.waitKey(1) & 0xFF == ord('q'):**

**cv2.destroyAllWindows()**

**camera.release()**

**break**

**Chapter 7**

**Conclusions**

A real-time eye blink detection algorithm was presented. We quantitatively demonstrated that Haar feature-based cascade classifiers and regression-based facial landmark detectors are precise enough to reliably estimate the positive images of face and a level of eye openness. While they are robust to low image quality (low image resolution in a large extent) and in-the- wild.

###### **Limitations:**

1. **Use of spectacles**: In case the user uses spectacle then it is difficult to detect the state of the eye. As it hugely depends on light hence reflection of spectacles may give the output for a closed eye as open eye .hence for this purpose the closeness of eye to the camera is required to avoid light.
2. **Multiple face problem :**

If multiple face arises in the window then the camera may detect morenumber of faces undesired output may appear. Because of different condition of different faces. So, we need to make sure that only the driver face come within the range of the camera. Also, the speed of detection reduces because of operation on multiple faces.

**Chapter 8**

**Future Work**

Our model is designed for detection of drowsy state of eye and give and alert signal or warning in the form of audio alarm. But the response of driver after being warned may not be enough to stop causing the accident meaning that if the driver is slow in responding towards the warning signal then accident may occur. Hence to avoid this we can design and fit a motor driven system and synchronize it with the warning signal so that the vehicle will slow down after getting the warning signal automatically.

We can also provide the user with an Android application which will provide with the information of his/her drowsiness level during any journey. The user will know Normal state, Drowsy State, the number of times blinked the eyes according to the number of frames captures. Which can be shown in Fig 15

****

***Fig 15 Android Application for Future scope***