

*A Project report on*

# **DRIVER'S DROWSINESS SYSTEM**

*Submitted in partial fulfillment of the requirements  
for the award of the degree of*

## **BACHELOR OF TECHNOLOGY**

*in*

**Computer Science & Engineering**

*By*

**N. PRAGATHI**

**164G1A0568**

**K. PRANEETHA**

**164G1A0569**

**B. RAVEENA SAI**

**164G1A0578**

**Y. SREEDHAR JASHWANTH**

**164G1A05A1**

**S. VIJAYKUMAR**

**164G1A05B7**

**Under the Guidance of**

**Mr. M. Narasimhulu, M.Tech.**

**Assistant Professor**



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**(B. Tech program accredited by NBA)**

**SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY: ANANTAPURAMU**

**(Accredited By NAAC With 'A' Grade, Affiliated To JNTUA, Approved By AICTE, New Delhi)**

**2019-2020**

**SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY: ANANTHAPURAMU**

(Accredited by NAAC with 'A' Grade, Affiliated to JNTUA, Approved by AICTE,  
New Delhi) Rotarypuram Village, B.K. Samudram Mandal, Ananthapuramu – 515701

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

(B. Tech Program accredited by NBA)



**Certificate**

This is to certify that the project report entitled **Driver's Drowsiness System** is the bonafide work carried out by **N.Pragathi** bearing Roll Number **164G1A0568**, **K.Praneetha** bearing Roll Number **164G1A0569**, **B.Raveena Sai** bearing Roll Number **164G1A0578**, **Y.Sreedhar Jaswanth** bearing Roll Number **164G1A05A1**, **S.Vijay Kumar**, bearing Roll Number **164G1A05B7** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2019-2020.

**Signature of the Guide**

Mr. M. Narasimhulu, M.Tech.  
Assistant Proffessor

**Head of the Department**

Dr. G. K. Venkata Narasimha Reddy, Ph.D  
Professor & HOD

Date:  
Place: Rotarypuram

EXTERNAL EXAMINER

## ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned my effort with success. It is a pleasant aspect that I have now the opportunity to express my gratitude for all of them.

It is with immense pleasure that I would like to express my indebted gratitude to my Guide **Mr. M. Narasimhulu, M.Tech. Assistant Professor, Computer Science and Engineering**, who has guided me a lot and encouraged me in every step of the seminar work. I thank him for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project work.

We express our deep felt gratitude to **Mr. P. Veera Prakash, M.Tech, (PhD), Assistant Professor**, project coordinator valuable guidance and unstinting encouragement enabled us to accomplish our project successfully in time.

We are very much thankful to **Dr. G. K. Venkata Narasimha Reddy, PhD, Professor and Head of Department, Computer Science & Engineering**, and all the faculty members of the department for their kind support and for providing necessary facilities and helpful information to carry out the work.

We wish to convey my special thanks to **Dr. T. Hitendra Sarma, Ph.D, Principal, Srinivasa Ramanujan Institute of Technology** for giving the required information in doing our project work. Not to forget, we thank all other faculty and non-teaching staff, and my friends who had directly or indirectly helped and supported me in completing our project in time.

We also express our sincere thanks to the Management for providing excellent facilities.

Finally, we wish to convey our gratitude to our family who fostered all the requirements and facilities that we need.

(N. Pragathi)

(K. Praneetha)

(B. Raveena Sai)

(Y. Sreedhar Jaswanth)

(S. Vijay Kumar)

## DECLARATION

We, N. Pragathi bearing reg no : 164G1A0568, K. Praneetha bearing reg no : 164G1A0569, B. Raveena Sai bearing reg no : 164G1A0578, Y. Sreedhar Jaswanth bearing reg no: 164G1A05A1 and S. Vijay Kumar bearing reg no: 164G1A05B7 students of SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY, Rotarypuram, hereby declare that the dissertation entitled “DRIVER’S DROWSINESS SYSTEM” embodies the report of our project work carried out by us during IV Year Bachelor of Technology under the guidance of Mr. M. Narasimhulu, M.Tech, Department of CSE and this work has been submitted for the partial fulfillment of the requirements for the award of Bachelor of Technology degree.

The results embodied in this project report have not been submitted to any other Universities or Institute for the award of Degree.

N. PRAGATHI	164G1A0568
K. PRANEETHA	164G1A0569
N. RAVEENA SAI	164G1A0578
Y. SREEDHAR JASWANTH	164G1A05A1
S. VIJAY KUMAR	164G1A05B7

# CONTENTS

	<b>Page No.</b>
<b>Abstract</b>	<b>viii</b>
<b>List of Figures</b>	<b>ix</b>
<b>List of Screens</b>	<b>xi</b>
<b>1. Introduction</b>	<b>1</b>
1.1 Motivation	1
1.2 Problem Definition	1
1.3 Objective of Project	1
1.4 Limitations of Project	2
1.5 Organization of Project	2
<b>2. Literature Survey</b>	<b>3</b>
2.1 Introduction	3
2.2 Existing System	3
2.3 Proposed System	5
2.4 Conclusion	5
<b>3. Analysis</b>	<b>6</b>
3.1 Introduction	6
3.2 Software Requirements Specification	6
3.3 Installation of Python	8
3.4 Installation of OpenCV	13
3.5 Hardware Requirements	17
<b>4. Design</b>	<b>19</b>
4.1 Introduction	19
4.2 Proposed Architecture	20
4.3 Description of Modules	21
4.4 UML Diagrams	21
4.5 Data Flow Diagram	26

<b>5. Implementation</b>	<b>29</b>
5.1 Introduction	29
5.2 Workflow of the proposed system	29
5.3 Coding	35
<b>6. Testing and Validation</b>	<b>47</b>
6.1 Introduction	47
6.2 Principles of Testing	47
6.3 Types of Software Testing	48
6.4 Test Cases	50
<b>Conclusion</b>	<b>51</b>
<b>References</b>	<b>52</b>

## **ABSTRACT**

A Driver Drowsiness Detection System is a real-time problem-solving system, which is capable of detecting the drowsiness or the fatigue of the driver while driving. The fatigue for the driver is caused due to the continuous long drives without any rest which apparently give rise to the accidents. According to a survey held, the maximum number of accidents have been taken place due to fatigue of the driver while driving to the long distances without any break. Here, in this paper we propose a system which ensures the safety of the diver by monitoring the status of the driver's eye and alerting the driver whenever necessary without fail. Also, we have two stages of alerting the driver, in the first stage of alert, we alert by focusing on the eye blink rate of driver where as in the second stage of alert, we go through a process which monitors the status of driver's eye indicating that the driver is drowsy when he falls asleep.



## **List of Figures**

<b>Fig No</b>	<b>Description</b>	<b>Page No</b>
3.3	Python open window	9
3.3.1	Python Download window	9
3.3.2	Python Versions window	10
3.3.3	Python Files window	10
3.3.4	Python Run window	11
3.3.5	Python Installation window	11
3.3.6	Python setup window	12
3.3.7	Python setup completed window	12
3.3.8	Python IDLE window	13
3.4	OpenCV installation window	13
3.4.1	Scipy installation window	14
3.4.2	Imutils installation window	14
3.4.3	Numpy installation window	15
3.4.4	Argparse installation window	15
3.4.5	Threading installation window	16
3.4.6	Dlib installation window	16
3.4.7	Time installation window	17
3.4.8	Playsound installation window	17
4.2	Proposed system Architecture	20

4.4.1	UseCase Diagram	23
4.4.2	Sequence Diagram	24
4.4.3	Collaboration Diagram	25
4.5	Recording driver's eye frames	27
4.5.1	Detection of eye features of driver	27
4.5.2	Data flow diagram of system	28
5.2.1	Plotting points on face	30
5.2.3	Conversion of RGB to Gray	31
5.2.4	EAR Calculation	32
5.2.5	Construction of convex hull contours	33
5.2.6.1	Hoe blinks and plotting EAR over time	33
6.3.1.1	Black box testing	48
6.3.2.1	White box testing	49

## **List of Screens**

<b>Fig No</b>	<b>Description</b>	<b>Page No</b>
5.2.6.2	Detecting blinks & blink count	34
5.2.7	Drowsiness detection alert	34

# **CHAPTER-1**

## **INTRODUCTION**

### **1.1 Motivation:**

We have done a survey on many problems in the field of Artificial Intelligence, as it is one of the most emerging technologies. In this, we notice that drowsiness detection is one of the major challenges in Artificial Intelligence. . In general, the driver fatigue accounts for 25 percent of accidents and approximately 60 percent of road accidents result in death or serious injury. In a study by the National Transportation Research Institute (NTSRB) in which 107 random car accidents had been selected, fatigue accounted for 58% of the all accidents. A main cause of fatigue is sleeplessness or insomnia. Driver's drowsiness is a major contributing factor in severe road accidents that claims thousands of lives every year.

We found a great rising in the trends of drowsiness detection systems, but still there is a space for further improvement in present measures of drowsiness detection. The use of intelligent systems in cars has developed considerably. These systems monitor and transmit the condition of the car and the driver. Smart cars which use software techniques to control engine speed, steering, transmission, break etc., has removed the quality of driving drastically. So to control these all we need some safe guard for accidents.

### **1.2 Problem Definition:**

The driver drowsiness detection system reduces the number of accidents, loss of property and loss of human life. It detects the drowsiness of a person and also it detects less blink rate of a person. It also takes the blink count weather the person is blinking or not.

### **1.3 Objective of Project:**

The main objective of this is to develop a drowsiness detection system by monitoring the eyes. Detection of fatigue involves the observation of eye movements and blink patterns. Based upon these observations, detection of drowsiness and less blink rate is done.

## **1.4 Limitations of Project:**

The following are some of the limitations of the proposed system.

- The system fails, if the automobile driver is wearing any kind of sunglasses.
- The system does not function if there is light falling directly on the camera.
- If multiple faces appear in window, then camera may detect more number of faces, then undesired output may appear.
- The system does not detect during night times.

## **1.5 Organization of Project:**

The webcam will be present in front of the driver. It takes the live video stream of the person. It detects the eyes of the person and calculates the eye aspect ratio of the person's eye and checks the drowsiness of the person's eye whether the eye is opened or closed. Then it detects the alarm based on the drowsiness factor and also based upon his blinks. The alert is made for the driver. Here in this we count the blinks of the eye and suggest whether he is awake or not. This is our first stage. In second stage it give alert when the person goes to sleeping position. The driver will alert at the last stage and he recues from the accident. When the person open his eyes and sleep then it will check for the blinks of the person if the person does not blink the for there seconds it will alert the person that he is sleepy. These all describe the process of the project and how it works.

## **CHAPTER-2**

### **LITERATURE SURVEY**

#### **2.1 Introduction:**

Drowsiness is one of the main factors that threaten the road safety and causes the severe injuries, deaths and economical losses. The increased drowsiness reduces the driving performance.

It has been observed that when the drivers do not take break they tend to run a high risk of becoming drowsy. Study shows that accidents occur due to sleepy drivers in need of a rest, which means that road accidents occurs more due to drowsiness rather than drink-driving. Attention assist can warn of inattentiveness and drowsiness in an extended range and notify drivers of their current state of fatigue. It is espically important in driving where time is a critical factor in driver's decision. On the other hand, another method to check the driver fatigue is monitoring the physical condition and facial expressions of the drivers, which wireless sensor networks are unab;e to process and transmit these information with adequate precision.

#### **2.2 Existing System:**

M. Hemamalini, P. Muhilan“Accident prevention using eye blink sensor”,vol 1, Issue L11, 2017. By using non-intrusive machine vision based concepts, drowsiness of the driver detected system is developed. Many existing systems require a camera which is involved in front of driver. It points straight towards to the face of the driver and monitors the driver's eyes in order to identify the drowsiness. For large vehicle such as heavy trucks and buses this arrangement is not pertinent. Bus has a large front glass window to have a broad view for save driving. If we place a camera on the window of the front glass, the camera blocks the frontal view of driver so it is not practical. If the camera is placed on the frame which is just above the window, then the camera is unable to detain the anterior view of the face of the driver correctly. The open CV detector detects only 40% of the face of driver in normal driving position in video recording of 10 minutes. In the oblique view, the open CV eye detector frequently fails to trace the pair of eyes. If the eyes are closed for five

successive frames the system concludes that the driver is declining slumbering and issues a warning signal. Hence this system is not applicable for large vehicles.

N.Mathiarasi, T.Sureshkumar from Department of Computer Science and Engineering has proposed the system named as Driver's Drowsiness and Unconsciousness Detection Methodologies. The main objective of this project is to detect driver's drowsiness and unconsciousness while on driving which helps to increase the vigilance of driver and to prevent the vehicle crashes on road due to driver's fatigue state. This uses various image processing techniques for detection of drowsiness like measuring of physiological signals, in vehicle sensors, computer vision methods and artificial neural networks. The limitations are artificial neural network based method gives the accurate results but it is difficult to implement in real time. The driver needs to wear it in his/her head continuously. Pulse rate method is most suitable one. Computerized visualization method needs more manual and computerized tracking but gives good results.

Gustavo A. Peláez C., Fernando García, Arturo de la Escalera, and José María Armingol," Driver Monitoring Based on Low-Cost 3-D Sensors. Previous studies have proposed a number of methods to detect drowsiness. After doing literature survey, different techniques has been found for detecting driver drowsiness and they use different types of data as input for their algorithm. After the survey of different types of methods, it is found that using camera is the best method which can be easily applied and appropriate in all conditions. We decide to explore this method of computer vision and proposed a noble method to detect driver drowsiness based on detecting eyelid closing and opening using artificial neural networks as classification algorithm. In this paper, First of all, the video frames are acquired from the camera which could be fixed in such a way that it should not obstruct the road-view of the driver. It is a serious highway safety problem. If drivers could be warned before they became too drowsy to drive safely, some of these crashes could be prevented. In order to reliably detect the drowsiness, it depends on the presentation of timely warnings of drowsiness. To date, the effectiveness of drowsiness detection methods has been limited by their failure to consider individual differences. Based on the type of data used, drowsiness detection can be conveniently separated into the two categories of intrusive and non-intrusive methods. During the survey, non-intrusive methods detect

drowsiness by measuring driving behavior and sometimes eye features, through which camera based detection system is the best method and so are useful for real world driving situations. This paper presents the review of existed drowsiness detection techniques that will be used in this system like Circular Hough Transform, FCM, Lab Color Space etc.

### **2.3 Proposed System:**

The proposed system will process the live video by mainly focusing and detecting the status of eyes and reducing the risk of accidents by alerting the driver quickly. This method uses 'OpenCV', 'NumPy' and other packages for the detection of drowsiness by processing only the eyes which has to be mainly focused and by converting the video into 'grey scale' to have a better performance.

### **2.4 Conclusion:**

This application can be implemented in the real time to reduce traffic accidents rate. It can also help drives to stay awake while driving by giving a warning when the driver is sleepy. It also alerts the driver if he blinks number of times. It provides users to help in reducing the accidents and be safe. This system specifies the future enhancement where the project has the scope to increase the alertness of the people while driving. This System is designed in such a way that it can be used in every vehicle currently on road to ensure the safety and reduce the chances of an accident due to drowsiness or distraction of driver.



## CHAPTER-3

### ANALYSIS

#### **3.1 Introduction:**

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure evaluates feasibility and risks associated with the project, and describe appropriate management and technical approaches. The most critical section of the project plan is a listing of high level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

After analyzing the requirements of the task to be performed, the next step is to analyze the problem and understand its context. The first activity in the phase is studying the existing system and other is to understand the requirements and domain of the new system. Both the activities are equally important, but the first activity serves as a basis of giving the functional specifications and then successful design of the proposed system. Understanding the properties and requirements of a new system is more difficult and requires creative thinking and understanding of existing running system is also difficult, improper understanding of present system can lead diversion from solution.

#### **3.2 Software Requirements Specification:**

A Software Requirements Specification (SRS) is a detailed description of a software system to be developed. The SRS is developed based the agreement between customer and contractors. It may include the use cases of how user is going to interact with software system. The software requirement specification document consistent of all necessary requirements required for project development. To develop the software system we should have clear understanding

of Software system. To achieve this we need to continuous communication with customers to gather all requirements.

A good SRS defines the how Software System will interact with all internal modules, hardware, communication with other programs and human user interactions with wide range of real life scenarios. Using the Software requirements specification (SRS) document, managers creates test plan. It is very important that testers must be cleared with every detail specified in this document in order to avoid faults in test cases and its expected results.

It is required to review or test SRS documents, before start writing test cases and making any plan for testing.

The Characteristics of an SRS are:

**Correctness of SRS should be checked:** Since the whole testing phase is dependent on SRS, it is very important to check its correctness. There are some standards with which we can compare and verify.

**Ambiguity should be avoided:** Sometimes in SRS, some words have more than one meaning and this might confuse testers making it difficult to get the exact reference. It is advisable to check for such ambiguous words and make the meaning clear for better understanding.

**Requirements should be complete:** When tester writes test cases, what exactly is required from the application, is the first thing which needs to be clear.

**Consistent Requirements:** The SRS should be consistent within itself and consistent to its reference documents. This should set to the standard and should be followed throughout the testing phase.

**Verification of expected result:** SRS should be clearly stated that what is expected since different testers would have different thinking aspects and may draw different results from this statement.

**Testing environment:** Some applications need specific conditions to test and also a particular environment for accurate result. SRS should have clear documentation on what type of environment is needed to set up.

**Pre-conditions defined clearly:** one of the most important part of test cases is pre-conditions. If they are not met properly then actual result will always be different expected result. Verify that in SRS, all the pre-conditions are mentioned clearly.

### **3.2.1 User Requirement:**

A description of each function required to solve the problem is presented in the functional description processing narrative is provided for each function, design constraints are stated and justified, performance characteristics are stated and one or more diagrams are included to graphically represent the overall structure of the software and interplay among software functions and other system elements.

The proposed system should be designed in such a way that should reliably detect drowsiness and also should also provide solution for detecting the less number of blinks so that it can serve its purpose as a system for promoting driver safety. These two constraints should be satisfied in order to reduce traffic accidents rate.

### **3.2.2 Software Requirement:**

The proposed system needs programming languages that are capable of performing image-processing techniques as well as mathematical operations like convolutions. The best suitable language that is flexible to perform the desired operations is Python. Since the proposed system is mostly dependent over tasks regarding video frames and their processing OpenCV module comes handy. The Programming language used here is Python 3.6.8 or any above versions. The Modules used here are OpenCV 3 or OpenCV 3.4 .We can use any Operating System like Windows 7/8/10.

## **3.3 Installation of Python:**

- Go to [www.python.org](http://www.python.org).
- Then, a window appears as shown below.

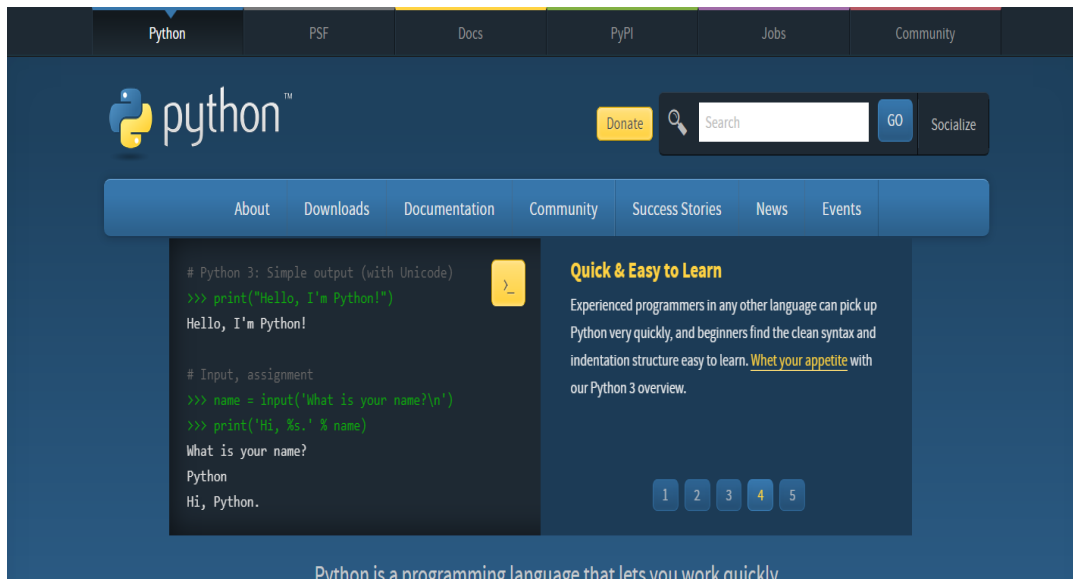


Fig 3.3.1 Python Open Window

- Click “Downloads” Link at the top of the page.
- Select “All Releases” and it will redirect to the download page.
- Then the page appears as given below.

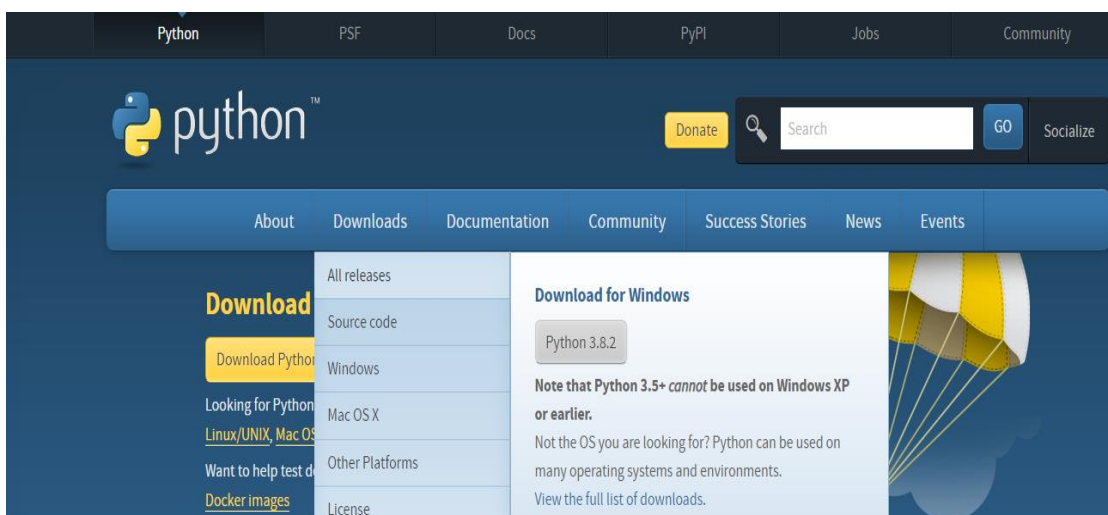
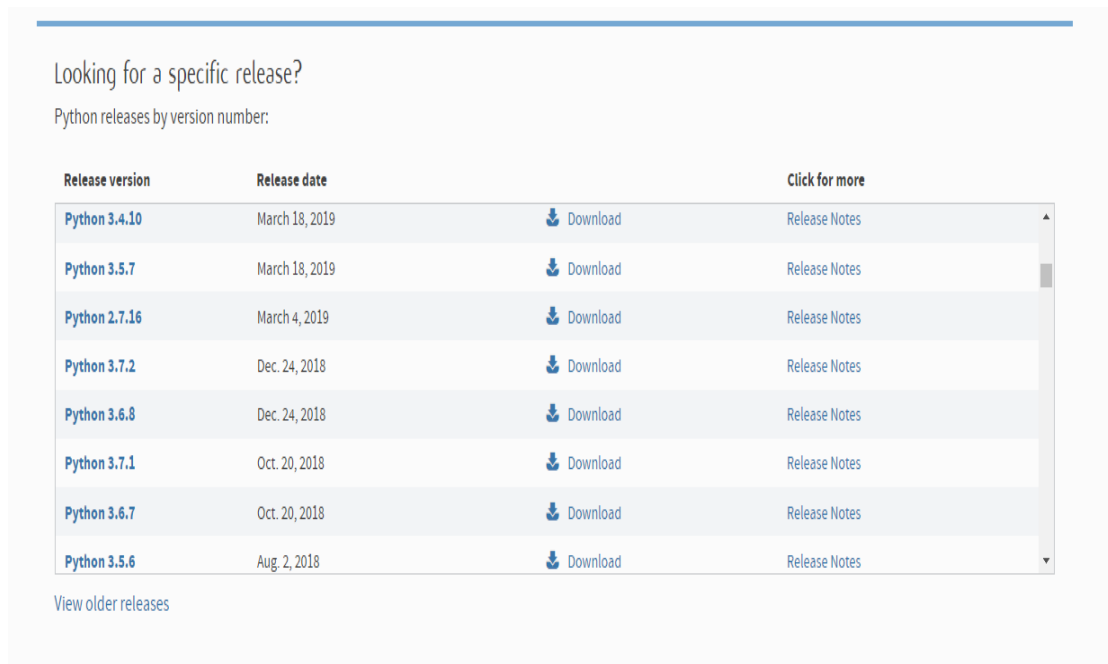


Fig 3. 3.2 Python Download Window

- Select the version Python 3.2 or above.



**Fig 3.3.3 Python Versions Window**

- After selecting the version it redirects to the version page that we have selected.
- There will be list of files and in that we should select Windows x86-64 executable installer for windows. The Selection of file depends upon the operating system that we have.

Version	Operating System	Description	MD5 Sum	File Size	GPG
<a href="#">Gzipped source tarball</a>	Source release		48f393a04c2e66c77bfc114e589ec630	23010188	<a href="#">SIG</a>
<a href="#">XZ compressed source tarball</a>	Source release		51aac91bdf8be95ec0a62d174890821a	17212420	<a href="#">SIG</a>
<a href="#">macOS 64-bit/32-bit installer</a>	Mac OS X	for Mac OS X 10.6 and later	eb1a23d762946329c2aa3448d256d421	33258809	<a href="#">SIG</a>
<a href="#">macOS 64-bit installer</a>	Mac OS X	for OS X 10.9 and later	786c4d9183c754f58751d52f509bc971	27073838	<a href="#">SIG</a>
<a href="#">Windows help file</a>	Windows		0b04278f5bdb8ee85ae5ae66af0430b2	7868305	<a href="#">SIG</a>
<a href="#">Windows x86-64 embeddable zip file</a>	Windows	for AMD64/EM64T/x64	73df7cb2f1500f36d7dbecac3968711	7276004	<a href="#">SIG</a>
<a href="#">Windows x86-64 executable installer</a>	Windows	for AMD64/EM64T/x64	72f37686b7ab240ef70fdb931bdf3cb5	31830944	<a href="#">SIG</a>
<a href="#">Windows x86-64 web-based installer</a>	Windows	for AMD64/EM64T/x64	39dde5f535c16d642e84fc7a69f43e05	1331744	<a href="#">SIG</a>
<a href="#">Windows x86 embeddable zip file</a>	Windows		60470b4cceba52094121d43cd3f6ce3a	6560373	<a href="#">SIG</a>
<a href="#">Windows x86 executable installer</a>	Windows		9c7b1ebdd3a8df0eebfda2f107f1742c	30807656	<a href="#">SIG</a>
<a href="#">Windows x86 web-based installer</a>	Windows		80de96338691698e10a935ecd0bdaacb	1296064	<a href="#">SIG</a>

**Fig 3.3.4 Python Files Window**

- After that it starts downloading the compatible version.
- Click on “Run” to run the file.

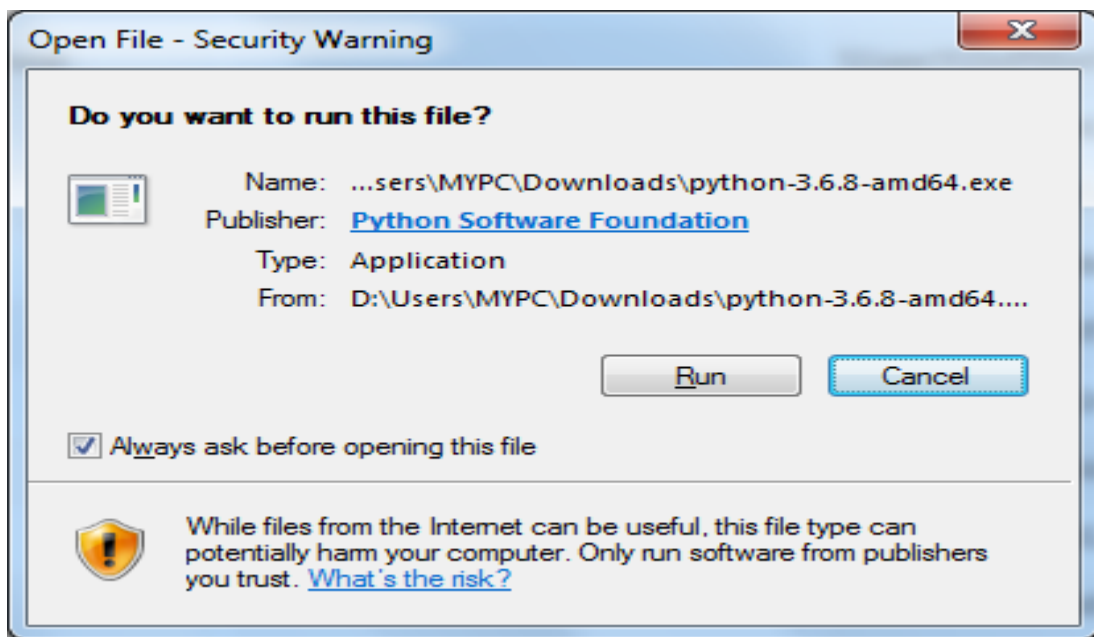


Fig 3.3.5 Python Run Window

- After clicking the “Run” button. Then installation window appears.

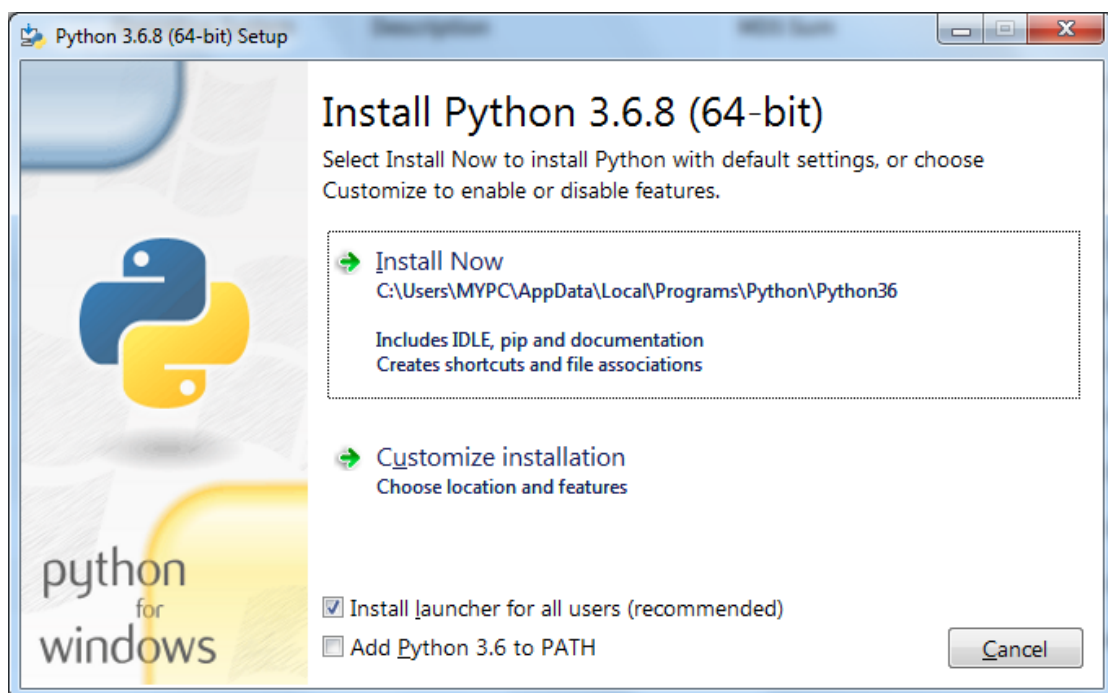
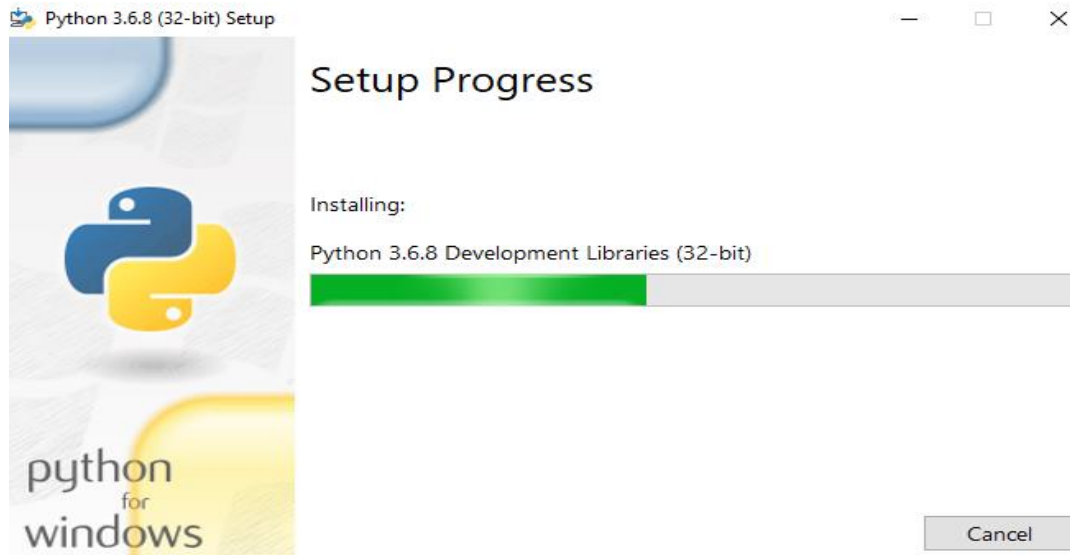


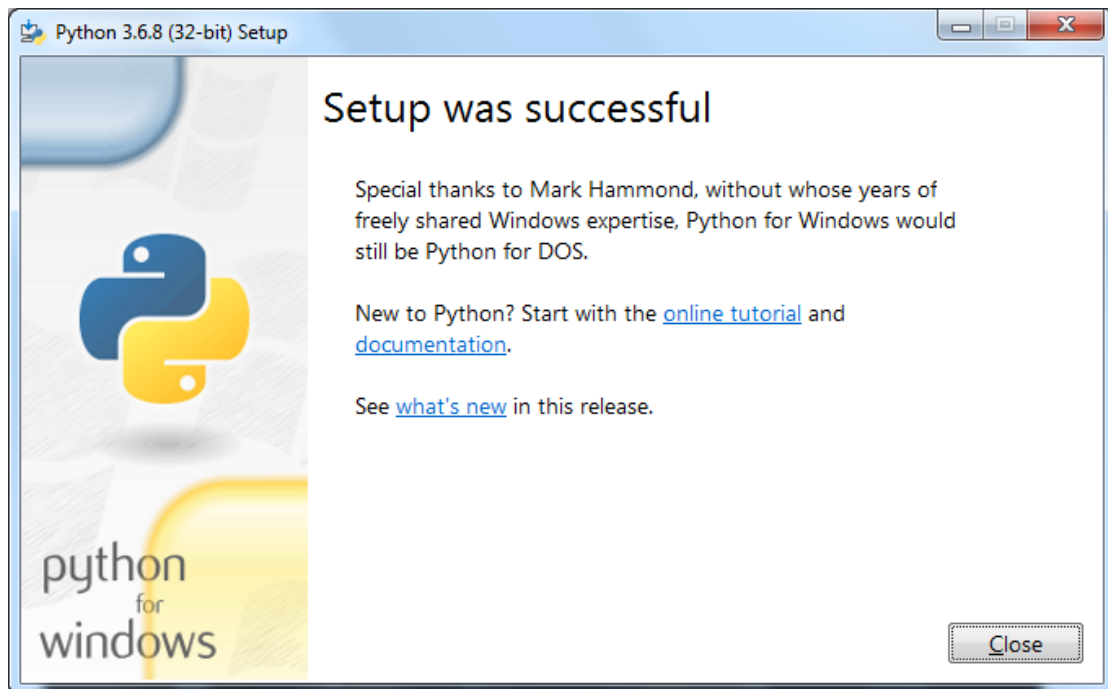
Fig 3.3.6 Python Installation Window

- Then we should select “Add Python 3.6 to PATH”.
- If we want to add any features we can select “customize installation” and select the features that we require.
- Click on “Install Now” and it starts installing the python.



**Fig 3.3.7 Python setup window**

- After the setup progress, the following below window appears.



**Fig 3.3.8 Python setup completed window**

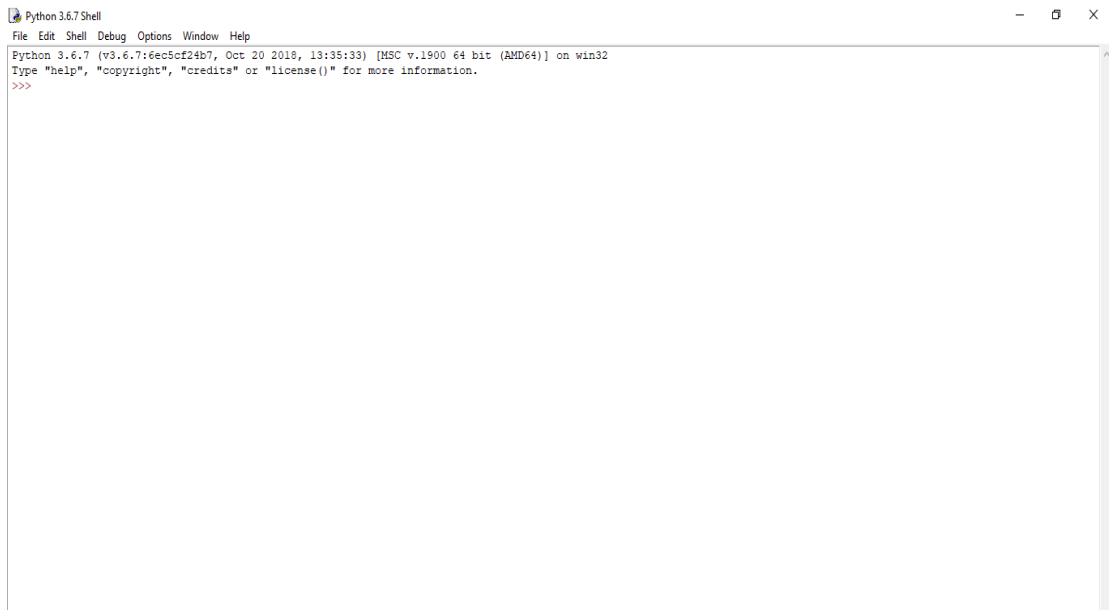
- After installation of the python you can launch it.

After the installation of python, you will be able to open IDLE( Integrated Development Environment (Python)) and run the python files.

To use IDLE editor on your computer, then the following steps should be done:

- Go to the text box at the lower left of your Windows desktop and type "IDLE".
- You should see a program come up from the search that says "IDLE (Python 3.6 32-bit) Desktop app". Click on this.

- You should get the “IDLE” Python shell, that looks like this:

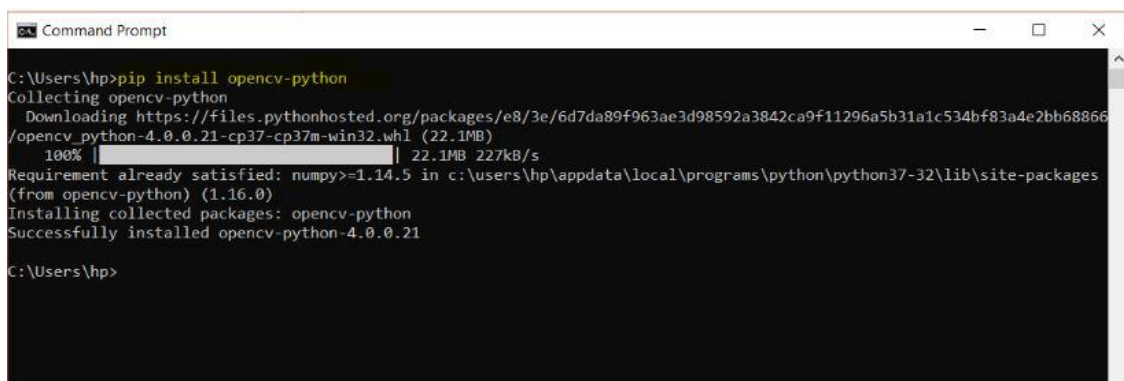


**Fig 3.3.9 Python IDLE window**

### 3.4 Installation of Open CV:

Now, you need dump into your system the Open CV module, for that :

- open your command prompt.
- Type the command “**pip install open CV-python**”.
- Then installation of Open CV gets started.
- If it gets installed successfully, the following should be the output.



**Fig 3.4 OpenCV Installation Completed Window**

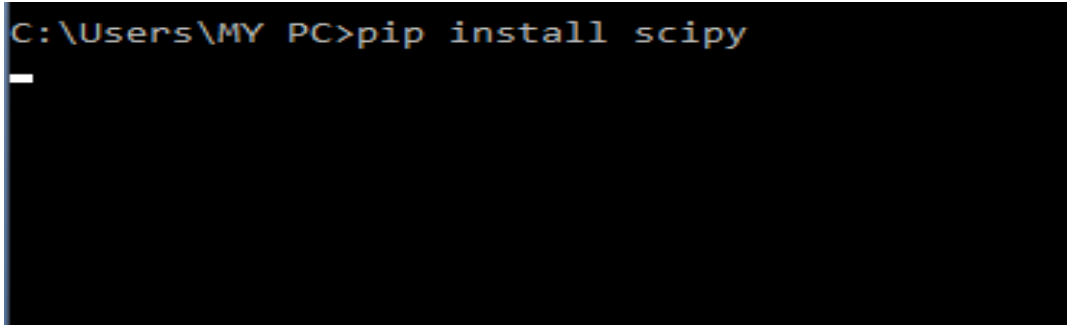


## Installation of Required Packages:

### 1.Scipy:

To install Scipy, follow the given steps:

- open your command prompt.
- Type the command “**pip install Scipy**”.
- Then installation of Scipy gets started.
- If installation starts, the following should be the output.



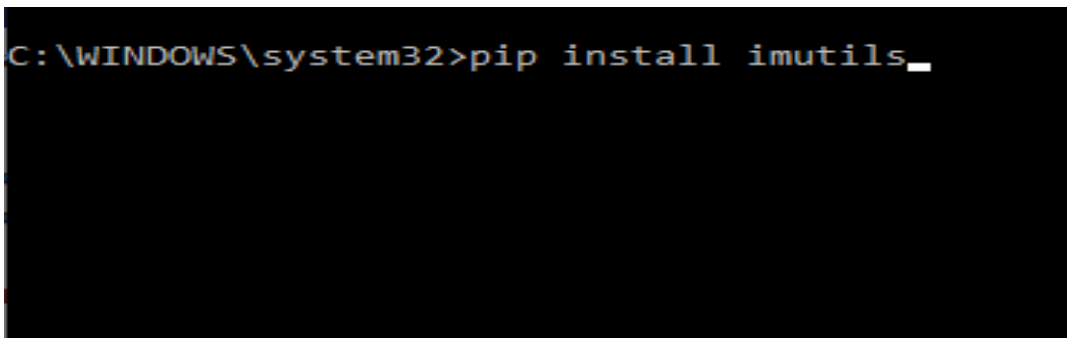
```
C:\Users\MY PC>pip install scipy
```

#### 3.4.1 Scipy Installation Window

### 2.imutils:

To install imutils, follow the given steps:

- open your command prompt.
- Type the command “**pip install imutils**”.
- Then installation of imutils gets started.
- If installation starts, the following should be the output.



```
C:\WINDOWS\system32>pip install imutils_
```

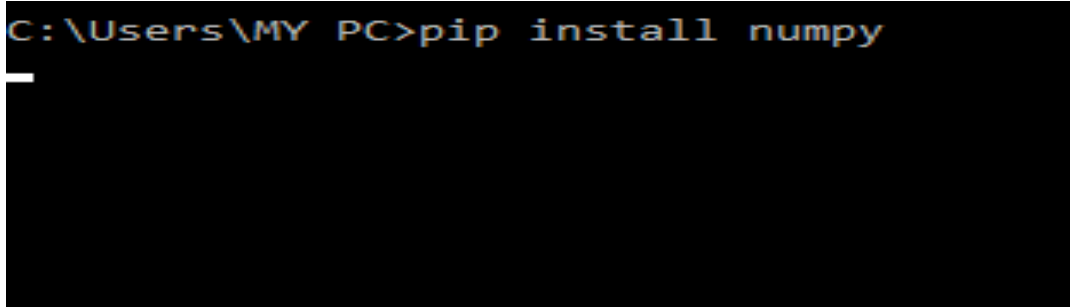
#### 3.4.2 imutils Installation Window

### 3.numpy:

To install numpy, follow the given steps:

- open your command prompt.
- Type the command “**pip install numpy**”.

- Then installation of numpy gets started.
- If installation starts, the following should be the output.



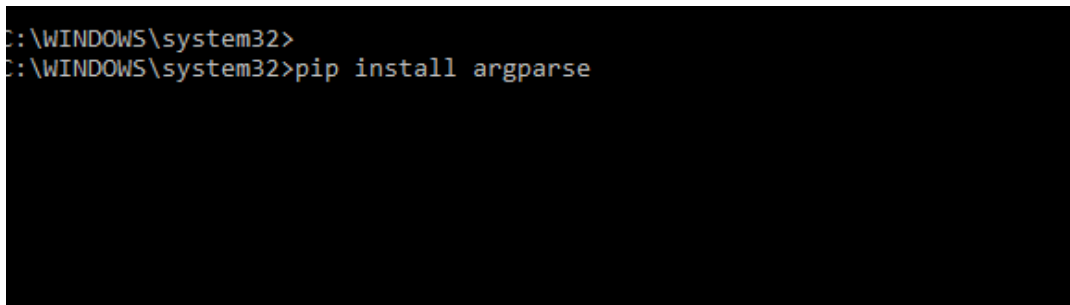
```
C:\Users\MY PC>pip install numpy
```

### 3.4.3 Numpy Installation Window

#### 4.argparse:

To install argparse, follow the given steps:

- open your command prompt.
- Type the command “**pip install argparse**”.
- Then installation of argparse gets started.
- If installation starts, the following should be the output.



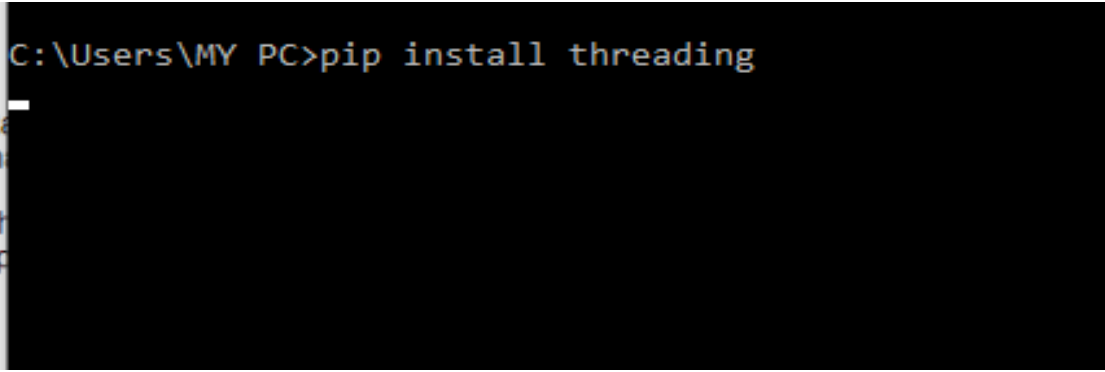
```
C:\WINDOWS\system32>
C:\WINDOWS\system32>pip install argparse
```

### 3.4.4 Argparse Installation Window

#### 5.Threading:

To install threading ,follow the given steps:

- open your command prompt.
- Type the command “**pip install thread**”.
- Then installation of thread gets started.
- If installation starts, the following should be the output.



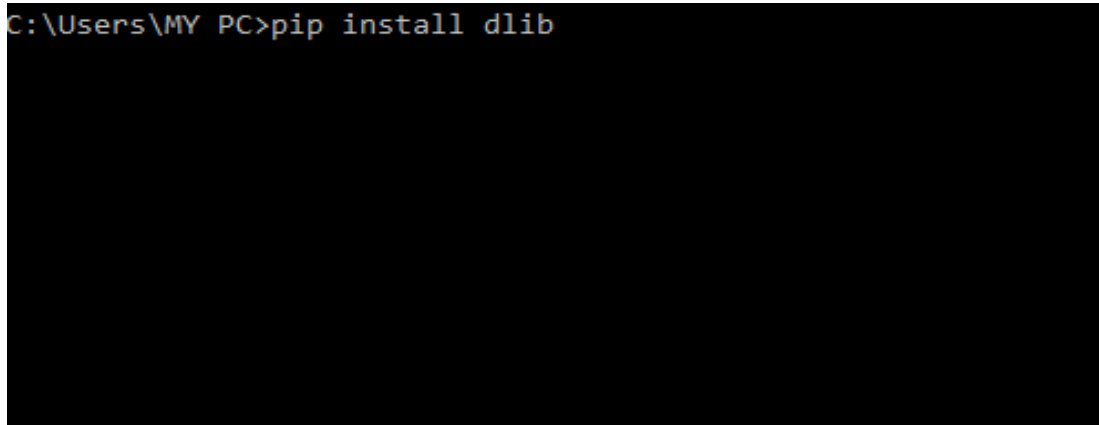
```
C:\Users\MY PC>pip install threading
```

### 3.4.5 Threading Installation Window

#### 6.dlib:

To install dlib, follow the given steps:

- open your command prompt.
- Type the command  
["https://files.pythonhosted.org/packages/0e/ce/f8a3cff33ac03a8219768f0694c5d703c8e037e6aba2e865f9bae22ed63c/dlib-19.8.1-cp36-cp36m-win\\_amd64.whl#sha256=794994fa2c54e7776659fddb148363a5556468a6d5d46be8dad311722d54bfct"](https://files.pythonhosted.org/packages/0e/ce/f8a3cff33ac03a8219768f0694c5d703c8e037e6aba2e865f9bae22ed63c/dlib-19.8.1-cp36-cp36m-win_amd64.whl#sha256=794994fa2c54e7776659fddb148363a5556468a6d5d46be8dad311722d54bfct)
- Then installation of dlib gets started.
- If installation starts, the following should be the output.



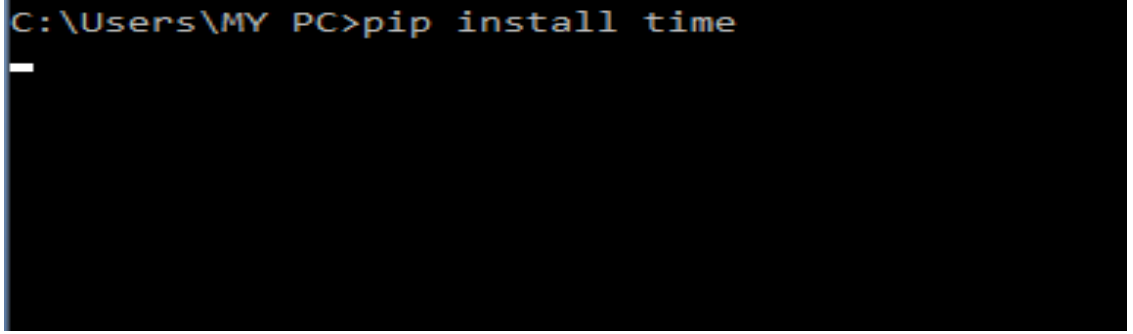
```
C:\Users\MY PC>pip install dlib
```

### 3.4.6 Dlib Installation Window

#### 7.time:

To install time, do the following steps:

- open your command prompt.
- Type the command **"pip install time"**.
- Then installation of time gets started.
- If installation starts, the following should be the output.



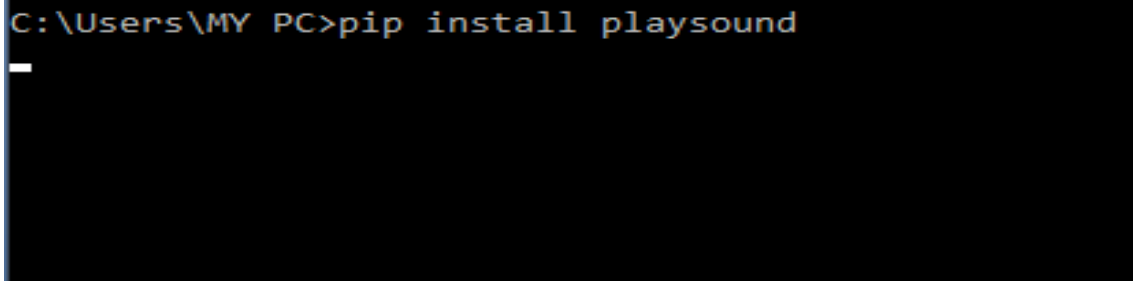
```
C:\Users\MY PC>pip install time
```

### 3.4.7 Time Installation Window

#### 8.playsound:

To install playsound,do the following steps:

- open your command prompt.
- Type the command “**pip install playsound**”.
- Then installation of playsound gets started.
- If installation starts, the following should be the output.



```
C:\Users\MY PC>pip install playsound
```

### 3.4.8 playsound Installation Window

## 3.5 Hardware Requirements:

Coming to the hardware requirements, when working with your own personal computer it would be necessary to have no other external hardware, but if you need work with the real-time video surveillance cameras, you need some external that is capable of processing a scripting language and also it should facilitate to get connected with external video source. Some of the examples of such integrated circuits are **Arduino, Raspberry Pi, and Beaglebone etc.** For a normal personal computer, since the necessary drivers for working of the camera are pre-installed so there is no need to separately install them you can directly work with the camera and use the necessary image-processing techniques with simple python scripting. In case of integrated circuits, all the necessary installations must be done previously to collaboratively work with all the components of the system. For example, when you work with Raspberry Pi in order to work with it connecting a video source you will

need a module called Pi Camera Module. So based upon the external circuit that is used necessary modules must be installed correspondingly. The RAM used here is 4GB and Higher and the processor used is Intel i3 and above. And Hard Disk used should be 320MB.

## CHAPTER-4

### DESIGN

#### 4.1 Introduction:

System design provides an understanding of the procedural details, necessary for implementing the system recommended in the feasibility study. Basically it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases. System design consists of three major steps and they are: Drawing of the expanded system data flow charts to identify all the processing functions required, The allocation of the equipment and the software to be used and the identification of the test requirements for the system.

In design phase, detailed design of the system is carried out and a user oriented performance specification is convenient for a technical design specification. Principle activities performed during design phase include the allocation of functions between computer programs, equipment and manual task, and test requirements definition.

Design phase begins with system design. This step involves allocation of function. Effective input design minimizes errors made by data entry operators. Output designs have been ongoing activity almost from the beginning of the project. Here the, layout for all the system outputs are prepared.

Input design is a part of overall system design, which. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

The principle behind a Drivers Drowsiness Detection is to detect physiological and/or performance indicators of driver drowsiness and then generates an alarm to the driver regarding his or her state of drowsiness. The purpose of providing such information to the driver in an impaired state is to motivate him or her to take corrective action.

## 4.2 Proposed Architecture:

The proposed system checks the availability of web camera and detection of eyes is done and then it will operate considering condition of driver's eyes. According to this architecture, the operation of the eye condition is made in such a way that it checks for less number of blinks and if the person blinks less number of times, the alarm gets generated and if not that case it again checks for blinks and after that at second stage, it checks for drowsiness factor, then if person is drowsy, then alarm gets generated if not that case control moves to check drowsiness factor. This architecture is mostly useful to avoid the accident occurrence. It also ensures the safety of the driver thereby generating the alarm and alerting the driver. The general diagram of system has been shown below in Fig.4.2.

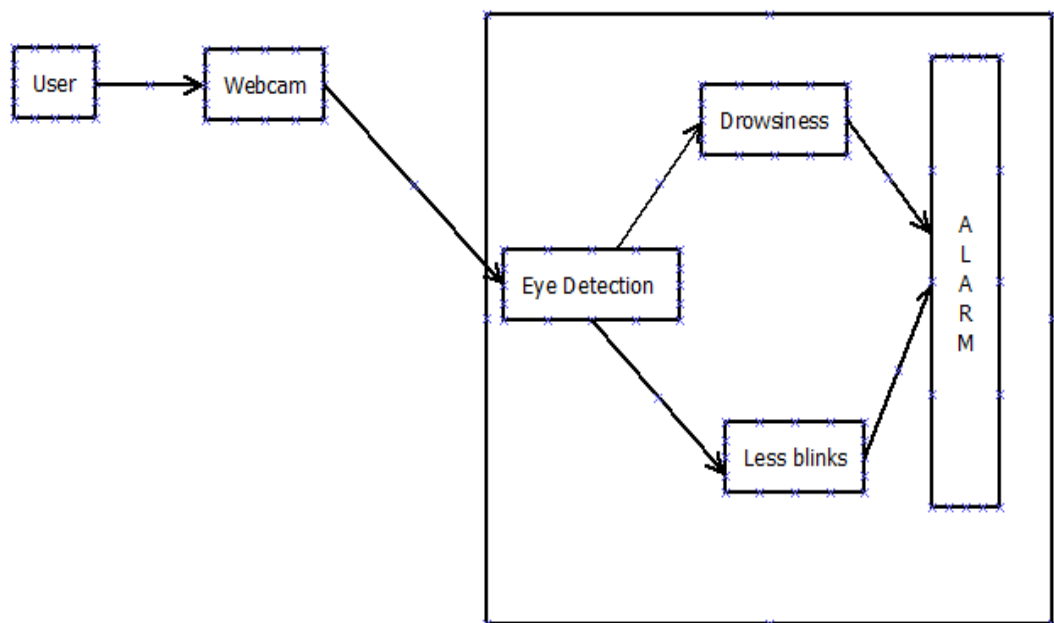


Fig 4.2 Proposed System Architecture

### 4.3 Description of Modules:

#### **Segmentation of eyes:**

The eye is segmented from the input video that is recorded by the camera will be given as inputs for segmenting the eyes.

#### **Eyes condition:**

The position of the driver's eye is determined by using appropriate threshold. In this work, detection of the eyes region is considered.

#### **Less Blinks Detection:**

The detection of less blinks can be done by considering the input and if the person blinks less number of blinks, then generation of alarm is made. This is the first stage of alerting.

#### **Drowsiness Detection:**

The detection of drowsiness can be done by checking the drowsiness factor and if it is greater than standard drowsiness factor then the alert is done. This is the second stage of alerting.

### 4.4 UML Diagrams:

Unified Modeling Language (UML) is a general purpose modeling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite similar to blueprints used in other fields of engineering.

UML is not a programming language; it is rather a visual language. We use UML diagrams to portray the behavior and structure of a system. UML helps software engineers, businessmen and system architects with modeling, design and analysis. The Object Management Group (OMG) adopted Unified Modeling Language as a standard in 1997. International Organization for Standardization (ISO) published UML as an approved standard in 2005.

The Unified Modeling language allows software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic, semantic and pragmatic rules.

A UML System is represented by five different views that describe the system from distinctly different perspectives. Each view is defined by set of diagram. Each Unified Modeling Language diagram is designed to let customers and developers



view a software system from a different perspective and in varying degrees of abstraction. UML diagrams are commonly created in visual modeling tools.

The reasons for using UML are:

- Complex applications need collaboration and planning from multiple teams and hence require a clear and concise way to communicate amongst them.
- Businessmen do not understand code. So UML becomes essential to communicate with non programmer's essential requirements, functionalities and processes of the system.
- A lot of time is saved down the line when teams are able to visualize processes, user interactions and static structure of the system.
- The primary goals in the design of the UML are it is a general-purpose modeling language that all modelers can use. It is meant to support good practices for design, such as encapsulation, separation of concerns. Provide users with a ready-to-use, expressive visual modeling language so they can develop and exchange meaningful models.

#### **4.4.1 Use Case Diagrams:**

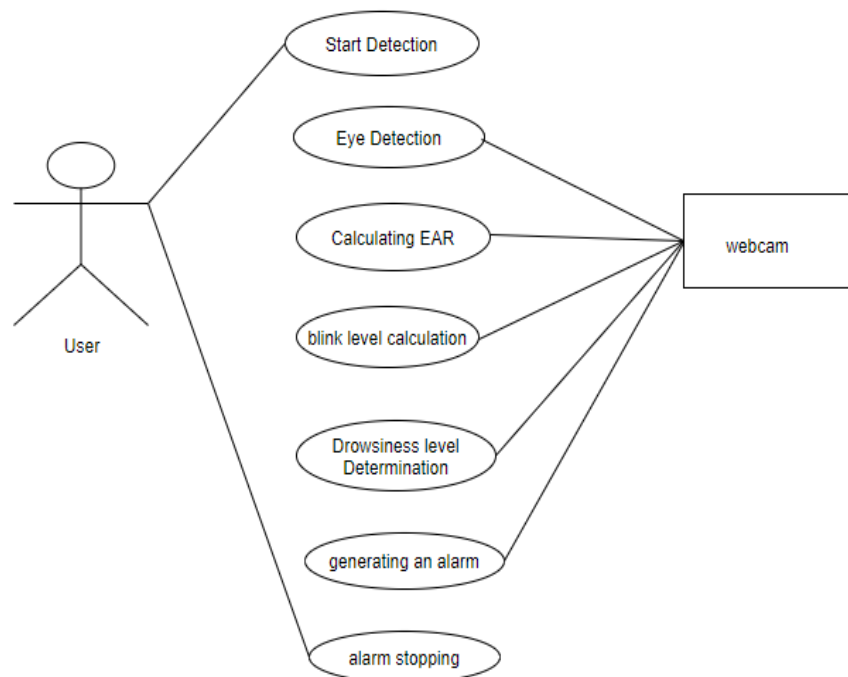
The Use case Diagram establishes the capability of the system as a whole. It is a methodology used in system analysis to identify, organize and clarify system requirements. It is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system.

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.

A use case defines the interactions between external actors and the system under consideration to accomplish a goal. Actors must be able to make decisions, but need not be human: "An actor might be a person, a company or organization, a computer program or computer system hardware, software, or both. Actors are always stakeholders, but not all stakeholders are actors, since they "never interact directly with the system, even though they have the right to care how the system behaves". For example, "the owners of the system, the company's board of directors,

and regulatory bodies such as the Internal Revenue Service and the Department of Insurance" could all be stakeholders but are unlikely to be actors.

In our proposed system there are two actors i.e. the user and webcam. The camera monitors users face and system records the live video in order to extract only the eye region and discard the surrounding region which we are not interested in. Then the conditions for drowsiness and less number of blinks are checked. If drowsiness or less number of blinks is detected then alarm is generated.

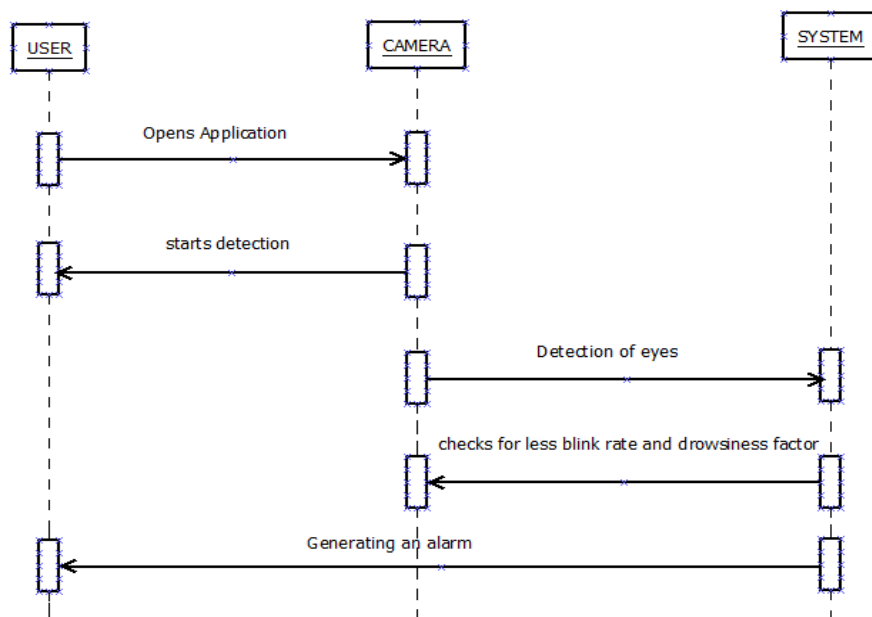


**Fig 4.4.1 Use Case Diagram**

#### **4.4.2 Sequence Diagram:**

A sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios. A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.



**Fig 4.4.2 Sequence Diagram**

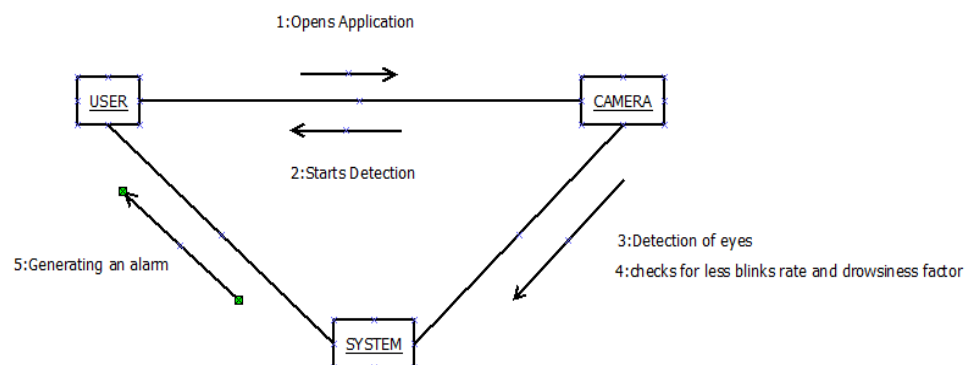
### 4.4.3 Collaboration Diagram:

Collaboration is a collection of named objects and actors with links connecting them. They collaborate in performing some task. It defines a set of participants and relationships that are meaningful for a given set of purposes. The Collaboration between objects working together provides emergent desirable functionalities in Object-Oriented systems. Each object (responsibility) partially supports emergent functionalities. Objects are able to produce (usable) high-level functionalities by working together. Objects collaborate by communicating (passing messages) with one another in order to work together

A collaboration Diagram is a sophisticated modeling tool can easily convert a collaboration diagram into a sequence diagram and the vice versa. A collaboration diagram resembles a flowchart that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time. A collaboration diagram is a interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Here the actions between various classes are represented by number format for the case of identification.

It first identifies the objects that participate in interaction and then establishes connection between the objects using links and finally adorns the links with the messages that object send and receive messages. It is used to model the flow of control.

Collaboration diagram has two features that are distinguished from sequence diagram. The first one is path where one object is linked to the another object. We can attach a path stereotype to the forend of a link. Another one is the sequence numbers that indicates type ordering of the messages.



**Fig 4.4.3 Collaboration Diagram of Proposed System**

## 4.5 Data Flow Diagram:

A data flow diagram (DFD) is a graphical representation of the flow of data through information system, modeling its process aspects. They can also be used for visualization of data processing.

A DFD shows what kinds of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.

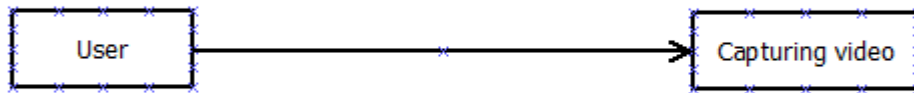
It is a way of representing a flow of a data of a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

There are several notations for displaying data-flow diagrams. For each data flow, at least one of the endpoints (source and / or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes.

When using UML, the activity diagram typically takes over the role of the data-flow diagram. To make the DFD more transparent (i.e. not too many processes), multi-level DFDs can be created. DFDs that are at a higher level are less detailed (aggregate more detailed DFD at lower levels). The contextual DFD is the highest in the hierarchy (see DFD Creation Rules). The so-called zero level is followed by DFD 0, starting with process numbering (e.g., process 1, process 2). In the next, the so-called first level - DFD 1 - the numbering continues. The number of levels depends on the size of the model system. DFD 0 processes may not have the same number of decomposition levels. DFD 0 contains the most important (aggregated) system functions. The lowest level should include processes that make it possible to create a process specification (Process Specification) for roughly one A4 page. If the mini-specification should be longer, it is appropriate to create an additional level for the process where it will be decomposed into multiple processes.

**Level 0:**

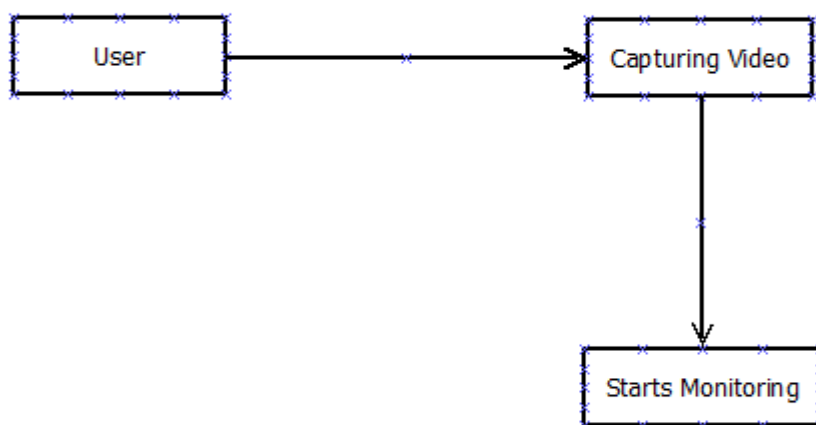
In this level, the person's face is captured by a webcam and the live streaming of video is done.



**Fig 4.5 Recording driver's eye frames**

**Level 1:**

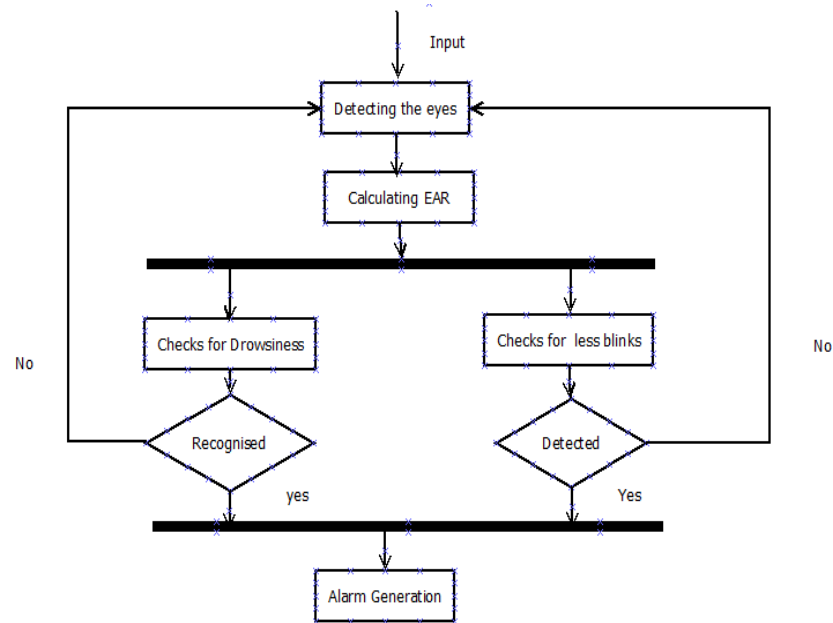
After the video is captured , the position of the eyes are detected. The eye movements are tracked.



**Fig 4.5.1 Detection of eye features of driver**

**Level 2:**

Here, after the monitoring, if the eye condition of the driver is not proper due to drowsiness and if blinking of eyes is less, an alarm is generated to alert the driver. The camera is positioned so that we can monitor the driver's eyes. We want to measure the movement of the eyes with the aid of the camera.



**Fig 4.5.2 Data Flow Diagram of the system**

## **CHAPTER-5**

### **IMPLEMENTATION**

#### **5.1 Introduction:**

The proposed system takes the video frames from source and processes it using image processing technique to effectively identify the eyes of a human in the video. The image processing-techniques are implemented in the OpenCV library, so by making use of it the most we are going to identify the eyes and contrast contour hulls around the eye to track status of the eye.

The techniques that are taken into consideration to create a perfect pipeline to perform the above task are Loading the facial land mark predictor converting the normal image to the gray scale image in order to reduce the dimensions of the image, calculating the Euclidean distance between the points around the eye hull and finding the EAR using the values obtained by calculating the Euclidean distance. After successful completion of contours, we are able to find the eye from that we can draw a conclusion that whether the driver is in the state of drowsiness or not.

#### **5.2 Workflow of the proposed system:**

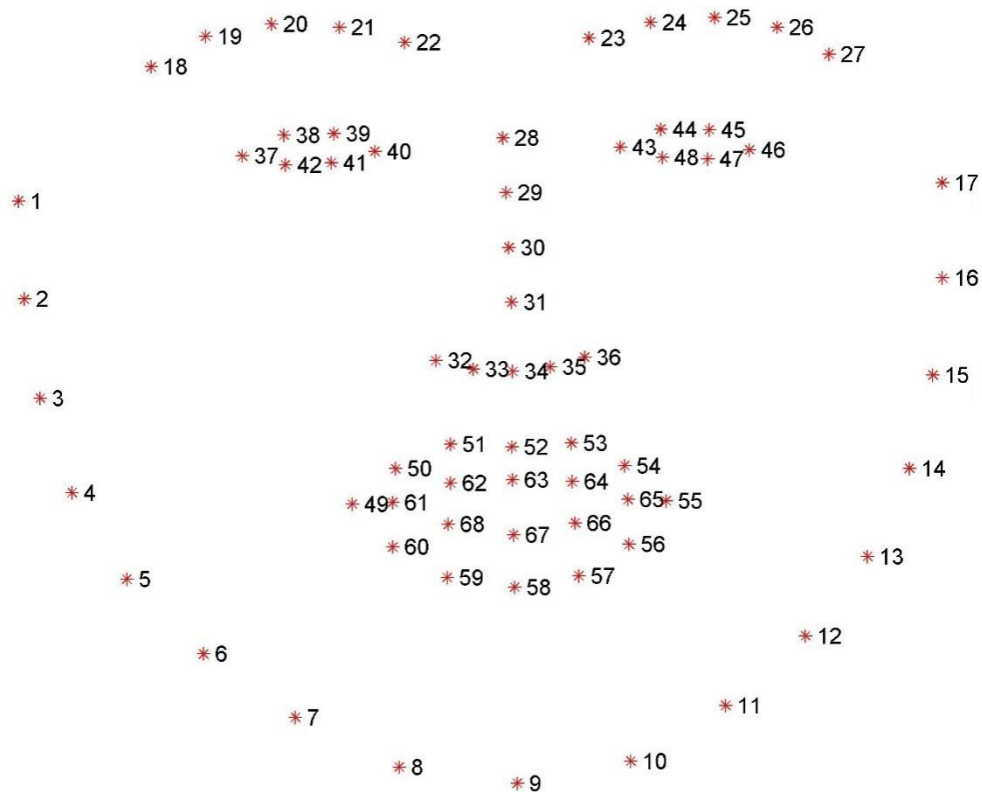
The following steps are involved in the proposed system to effectively identify the eyes and tracking the status of the eyes and to detect the drowsiness of the driver which results in a safe driving

- Loading the facial landmark predictor
- Capturing the frame
- BGR to Gray conversion
- Calculating the Euclidean distance& EAR
- Construction of contours
- Detecting blinks
- Detecting Drowsiness



### 5.2.1 Loading the facial landmark predictor:

It is produced by dlib are indexable list, it is implemented insidedlib produces the 68(x,y)-coordinates that map to specific facial structures. These 68 point mappings were obtained by training a shape predictor on the labeled “*IBUG 300-W dataset*”



**Fig5.2.1 Plotting the points of face**

```
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
```

```
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
```

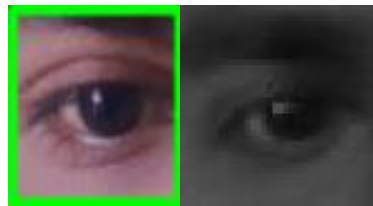
We use only these two lines to track the status of the eye which has to be focused for the detection of drowsiness of the driver.

### 5.2.2 Capturing the frames:

We have to capture the video from a video source. We are using the open source library namely, OpenCV to capture the video frames. It successfully capturing the video frame.

### 5.2.3 BGR to Gray Conversion:

RGB image contains lots of data which may not be required for your processing. Upon converting into gray scale it discards lots of information which are not required for processing. Generally, a RGB image is represented by 3 channels, therefore lots of data to be stored and to be manipulated, so to overcome this problem we are going to convert it into grayscale image i.e., we are going to average all the values in the 3 channels and store the value in one of the corresponding shades of gray, so that data gets reduced in the sense that the dimensions are reduced which in result reduces the computing time cost. In cv2, there is a function called `cvtColor()` used to convert an image from one color space to another. There are many more than 150 color-space conversion methods available in OpenCV.



**Fig..5.2.3 Conversion from RGB to Gray**

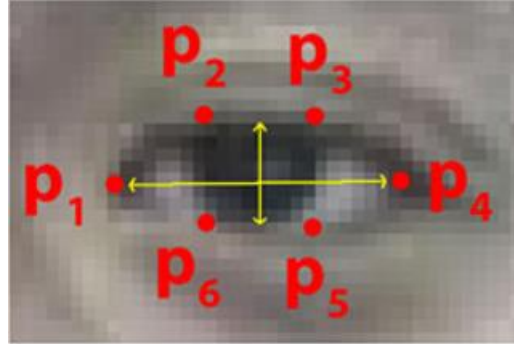
### 5.2.4 Calculating the Euclidean distance and EAR:

We have to calculate the Euclidean distance because the contours have to be shaped in hull shape for efficiency by only focusing on the area we needed. We are constructing the contours around the eye in hull shape by calculating the Euclidean distance between the points as shown by the formula “*dist.euclidean(eye[1],eye[2])*”

We also have to calculate the EAR i.e., *Eye Aspect Ratio*,

It is done because we have to focus on both the eyes as well, we will come across the different values for each eye so, as to maintain the accuracy

$$EAR = (\text{dist.euclidean}(\text{eye}[2], \text{eye}[6]) + \text{dist.euclidean}(\text{eye}[3], \text{eye}[5])) / 2.0 * \text{dist.euclidean}(\text{eye}[1], \text{eye}[4])$$



**Fig 5.2.4, EAR Calculation**

Final EAR:

it is the value obtained by the average of left EAR and right EAR

$$ear = (\text{leftEAR} + \text{rightEAR}) / 2.0$$

### **5.2.5 Construction of contours:**

Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition. Basically, the contours are in square shaped and we can change it according to our convenience. By the help of facial landmark predictor, we are able to spot the position of eye and draw contours around it. As processing an image from a square shaped contour is not necessary, we congest the contours in to a convex hull shape around the eye so that we get the only eye area we needed.



Fig5.2.5, Construction of convex hull contours

### 5.2.6 Detecting blinks:

As we have two stages of alerting the driver, this comes first by detecting the blink of a driver and counting them for every minute. If the count of that is less than the threshold given the alert will be raised indicating the drowsiness of driver.

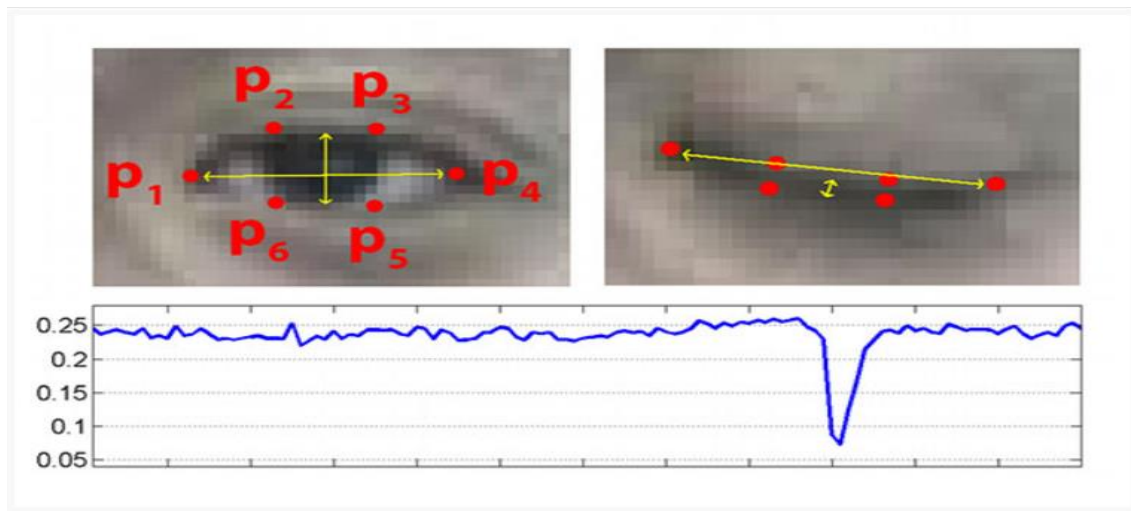
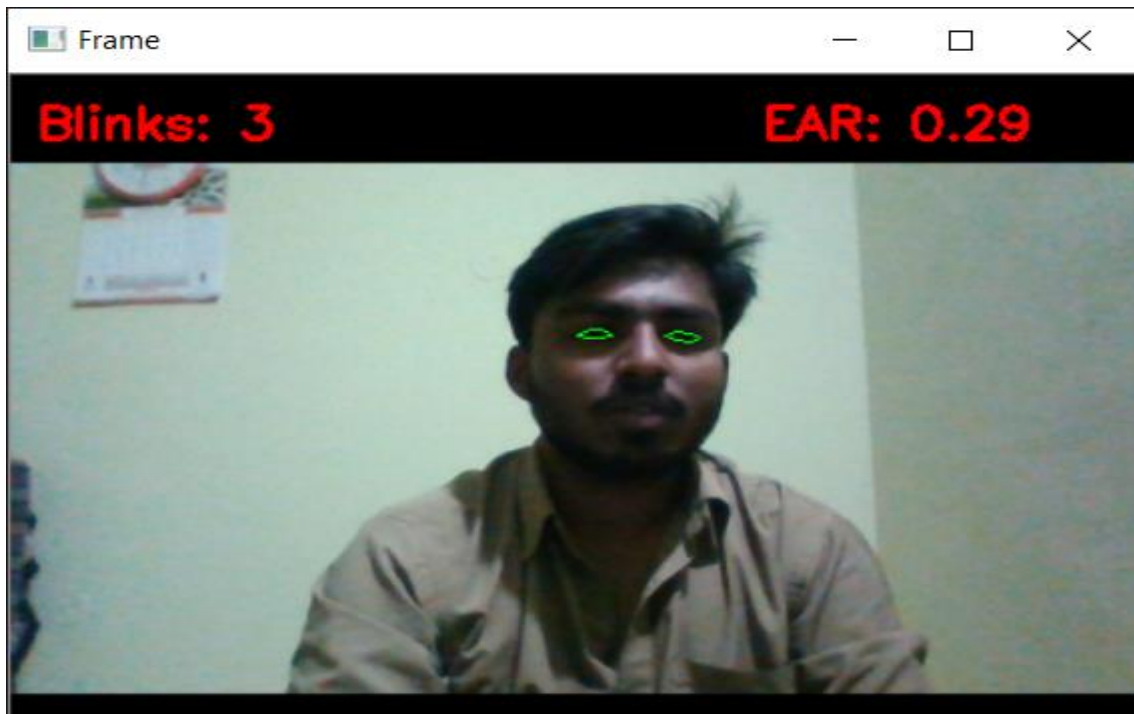


Fig 5.2.6.1, hoe blinks and plotting EAR over time.



**Fig5.2.6.2,( Detecting blinks &blink count)**

### **5.2.7 Drowsiness detection:**

As the flow is done with the first stage of alerting the script enters into the other level of alerting that is by calculating the drowsiness factor, if the resultant EAR is less than the drowsiness factor, the second alert comes into role.



**Fig 5.2.7,( Drowsiness detection & alert)**

## 5.3 Coding

Importing our required python packages.

### **blink\_detection.py:**

```
#import necessary packages

from scipy.spatial import distance as dist

from imutils.video import FileVideoStream

from imutils.video import VideoStream

from imutils import face_utils

import numpy as np

import argparse

import imutils

import time

import dlib

import cv2

from datetime import datetime
```

### **eye\_aspect\_ratio function:**

```
def eye_aspect_ratio(eye):

# compute the euclidean distances between the two sets of vertical eye landmarks (x,
y)-coordinates

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
```

### **count\_blinks( )function:**

### **Considering the local time:**

```
minute = int(datetime.now().strftime('%S'))
```

### **Defining EYE\_AR\_THRESH:**

```
EYE_AR_THRESH = 0.23

EYE_AR_CONSEC_FRAMES = 3

COUNTER = 0

TOTAL = 0
```

### **Facial landmark Predictor:**

```
print("[INFO] loading facial landmark predictor...")

detector = dlib.get_frontal_face_detector()

predictor = dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')
```

### **Extracting the Eye Regions:**

```
# grab the indexes of the facial landmarks for the left and

# right eye, respectively

(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]

(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
```

**Instantiate Video Stream:**

```
# start the video stream thread

    print("[INFO] starting video stream thread...")

# vs = FileVideoStream(args["video"]).start()

# fileStream = True

    vs = VideoStream(src=0).start()

# vs = VideoStream(usePiCamera=True).start()

    fileStream = False

    time.sleep(1.0)

# loop over frames from the video stream

    while True:

        # if this is a file video stream, then we need to check if there any more frames left in
        the buffer to process

            if fileStream and not vs.more():

                break

# grab the frame from the threaded video file stream, resize it, and convert it to
grayscale channels

    frame = vs.read()

    frame = imutils.resize(frame, width=450)

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

# detect faces in the grayscale frame

    rects = detector(gray, 0)
```



**Facial Landmark Detection to localize each of the important regions of face:**

```
# loop over the face detections
```

```
    for rect in rects:
```

```
# determine the facial landmarks for the face region, then convert the facial landmark
(x, y)-coordinates to a NumPy array
```

```
        shape = predictor(gray, rect)
```

```
        shape = face_utils.shape_to_np(shape)
```

```
# extract the left and right eye coordinates, then use the coordinates to compute the
eye aspect ratio for both eyes
```

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

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

```
        leftEAR = eye_aspect_ratio(leftEye)
```

```
        rightEAR = eye_aspect_ratio(rightEye)
```

```
# average the eye aspect ratio together for both eyes
```

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

**Visualize each of the Eye Regions:**

```
# compute the convex hull for the left and right eye, then
```

```
# visualize each of the eyes
```

```
        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)
```

**Check to see if person in video stream is starting to blink less number of times:**

# check to see if the eye aspect ratio is below the blink threshold, and if so, increment the blink frame counter

if ear < EYE\_AR\_THRESH:

COUNTER += 1

# otherwise, the eye aspect ratio is not below the blink threshold

else:

# if the eyes were closed for a sufficient number of then increment the total number of blinks

if COUNTER >= EYE\_AR\_CONSEC\_FRAMES:

TOTAL += 1

# reset the eye frame counter

COUNTER = 0

**Displaying the Output Frame:**

cv2.putText(frame,"Blinks:{ }".format(TOTAL),(10,30),cv2.FONT\_HERSHEY\_SIMPLEX, 0.7, (0, 0, 255), 2)

cv2.putText(frame,"EAR:{:.2f}".format(ear),(300,30),cv2.FONT\_HERSHEY\_SIMPLEX, 0.7, (0, 0, 255), 2)

# show the frame

cv2.imshow("Frame", frame)

key = cv2.waitKey(1) & 0xFF

# condition for how long does it take input

if abs(minute - int(datetime.now().strftime('%S'))) == 30:

```
        break

    # do a bit of cleanup

    cv2.destroyAllWindows()

    vs.stop()

    return TOTAL
```

### **detect\_drowsiness.py:**

```
#import necessary packages

from scipy.spatial import distance as dist

from imutils.video import VideoStream

from imutils import face_utils

from threading import Thread

import numpy as np

import playsound

import argparse

import imutils

import time

import dlib

import cv2
```

### **Sound Alarm:**

```
def sound_alarm(path):

    # play an alarm sound

    playsound.playsound(path)
```

**eye\_aspect\_ratio function:**

```
def eye_aspect_ratio(eye):  
  
    # compute the euclidean distances between the two sets of vertical eye landmarks (x,  
    y)-coordinates  
  
        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
```

**detect\_drowsiness( )function:****Defining EYE\_AR\_THRESH:**

```
EYE_AR_THRESH = 0.3  
  
EYE_AR_CONSEC_FRAMES = 48  
  
COUNTER = 0  
  
ALARM_ON = False
```

**Facial landmark Predictor:**

```
print("[INFO] loading facial landmark predictor...")  
  
detector = dlib.get_frontal_face_detector()  
  
predictor = dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')
```

**Extracting the Eye Regions:**

```
# grab the indexes of the facial landmarks for the left and  
  
# right eye, respectively  
  
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]  
  
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
```

**Instantiate Video Stream:**

```
print("[INFO] starting video stream thread...")  
  
vs = VideoStream(src=0).start()  
  
fileStream = False  
  
time.sleep(1.0)  
  
# loop over frames from the video stream  
  
while True:  
  
# grab the frame from the threaded video file stream, resize it, and convert it to  
# grayscale channels  
  
frame = vs.read()  
  
frame = imutils.resize(frame, width=450)  
  
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)  
  
# detect faces in the grayscale frame  
  
rects = detector(gray, 0)
```

**Facial landmark detection to localize each of the important regions of face:**

```
# loop over the face detections  
  
for rect in rects:
```

# determine the facial landmarks for the face region, then convert the facial landmark (x, y)-coordinates to a NumPy array

```
shape = predictor(gray, rect)
```

```
shape = face_utils.shape_to_np(shape)
```

# extract the left and right eye coordinates, then use the coordinates to compute the eye aspect ratio for both eyes

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

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

```
leftEAR = eye_aspect_ratio(leftEye)
```

```
rightEAR = eye_aspect_ratio(rightEye)
```

# average the eye aspect ratio together for both eyes

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

### **Visualize each of the Eye Regions:**

# compute the convex hull for the left and right eye, then

# visualize each of the eyes

```
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)
```

### **Check to see if person in video stream is starting to show symptoms of drowsiness:**

# check to see if the eye aspect ratio is below the blink

# threshold, and if so, increment the blink frame counter

```
if ear < EYE_AR_THRESH:

    COUNTER += 1

# if the eyes were closed for a sufficient number of frames then sound the alarm

if COUNTER >= EYE_AR_CONSEC_FRAMES:

    # if the alarm is not on, turn it on

    if not ALARM_ON:

        ALARM_ON = True

# check to see if an alarm file was supplied, and if so, start a thread to have the alarm

# sound played in the background

if "alarm.wav" != "":

    t = Thread(target=sound_alarm,args=("alarm.wav",))

    t.daemon = True

    t.start()

# draw an alarm on the frame

cv2.putText(frame,"DROWSINESSALERT!",(10,30),cv2.FONT_HERSHEY_
_SIMPLEX, 0.7, (0, 0, 255), 2)

# otherwise, the eye aspect ratio is not below the blink threshold, so reset the counter
and alarm

else:

    COUNTER = 0

    ALARM_ON = False
```

### **Displaying the output frame:**

```
# draw the computed eye aspect ratio on the frame to help with debugging and setting
the correct eye aspect ratio
```

# thresholds and frame counters

```
cv2.putText(frame,"EAR:{:.2f}".format(ear),(300,30),cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
```

# show the frame

```
cv2.imshow("Frame", frame)
```

```
key = cv2.waitKey(1) & 0xFF
```

# if the `q` key was pressed, break from the loop

```
if key == ord("q"):
```

```
    break
```

# do a bit of cleanup

```
cv2.destroyAllWindows ()
```

```
vs. stop()
```

```
return COUNTER
```

## **main.py**

**#importing the required files**

```
from blink_detection import count_blinks
```

```
from detect_drowsiness import detect_drowsiness
```

**#importing packages**

```
import playsound
```

```
l = []
```

**Checking the condition:**

```
while True:
```

**Checking for less number of blinks:**



```
if count_blinks() < 15:
```

```
    print('*'*5 + ' ALERT ' + '*'*5)
```

```
    playsound.playsound('C:/Users/MYPC/Desktop/final code/alarm.wav')
```

### **Checking for drowsiness:**

```
if detect_drowsiness() > 3:
```

```
    print('*'*5 + ' ALERT ' + '*'*5)
```

```
    playsound.playsound('C:/Users/MYPC/Desktop/final code/alarm.wav')
```

```
    break
```

## CHAPTER-6

### TESTING AND VALIDATION

#### 6.1 Introduction:

Testing is the process of evaluating a system or its components with the intent to find whether it satisfies the specified requirements or not.

Testing is a process of executing a program with an aim of finding errors. To make our software perform well it should be error free. If testing is done successfully it will remove all the errors from the software. Testing is a process of analyzing a software item to detect the differences between existing and required conditions and to evaluate the features of the software item.

#### 6.2 Principles of Testing

- All the test should meet the customer requirements.
- To make our software testing should be performed by a third party.
- All the test to be conducted should be planned before implementing it.
- It follows pareto rule i.e., 80/20 which is 80% of errors comes from 20% of program components.
- Start testing with small parts and extend it to large parts.
- Testing depends on the process and the associated stakeholders of the projects. Testing is done in different forms at every phase of SDLC.
- During the requirement gathering phase, the analysis and verification of requirements are also considered as testing.
- Reviewing the design in the design phase with the intent to improve the design is also considered as testing.
- Testing performed by a developed on completion of the code is also categorized as testing.

It is difficult to determine when to stop testing, as testing is a never ending process and no one can claim that a software is 100% tested.

Testing can be stopped in the following aspects such as

1. Testing deadlines
2. Completion of test case execution
3. Completion of functional and code coverage to a certain point
4. Bug rate falls below a certain level

The software testing is defined as-

It is the process in which the defects are identified, isolated, and subjected for rectification and finally make sure that all the defects are rectified , in order to ensure that the product is a Quality product.

The Objectives of Software Testing are-

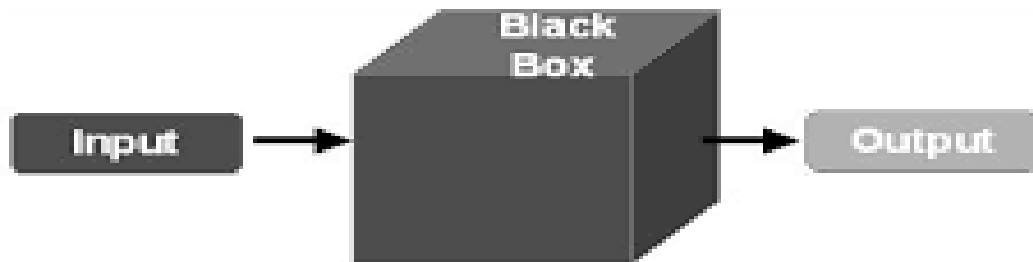
- Understand the difference between verification and validation testing activities.
- Understand what benefits the V model offers over other models.
- Be aware of other models in order to compare and contrast.
- Understanding the cost of fixing faults increases as you move the product towards live use.
- Understand what constitutes a master test plan in Software Testing.
- Understand the meaning of each testing stage in Software Testing.

### 6.3 Types of Software Testing

There are different types of Software Testing methods:poo0

#### 6.3.1 Black Box Testing:

It is also known as Behavioral testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.



**Fig 6.3.1.1 Black box testing**

This method is named because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. This method attempts to find errors in the following errors:

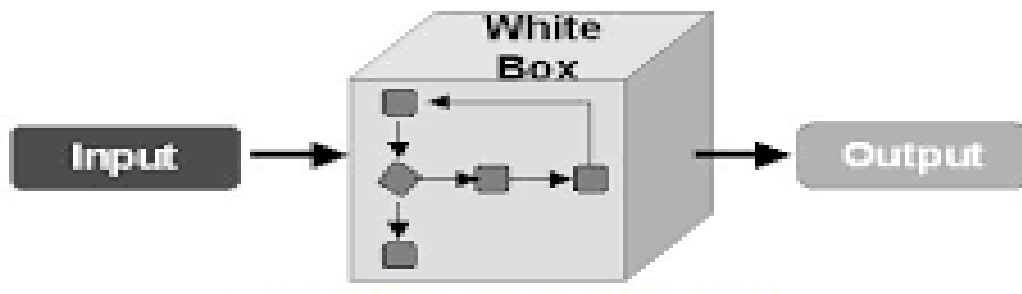
1. Incorrect or missing functions
2. Interface errors
3. Errors in data structures or external database

Some techniques that can be used for designing black box tests

- **Equivalence Partitioning:** It is a software test design technique that involves dividing input values into valid and invalid types and selecting values from each type for test data.
- **Boundary Value Analysis:** It involves the determination of boundaries for input values and selecting values that are at boundaries and just inside/ outside of the boundaries test data.
- **Cause-Effect Graphing:** It is a software test design technique that involves identifying the cases and effects producing a Cause-effect graph, and generating test cases accordingly.

### 6.3.2 White Box Testing:

It is also known as Clear Box Testing, Open Box Testing, Glass Box Testing, Transparent Box Testing, Code-Based Testing or Structural Testing. In this testing the internal structure/design/implementation of the item being tested is known as tester.



**Fig 6.3.2.1 White box testing**

Some Techniques involved in White Box Testing :

- **Statement Coverage:** It is aimed to exercise all programming statements with minimal tests.
- **Branch Coverage:** It runs a series of tests to ensure that all branches are tested only once.
- **Path Coverage:** It corresponds to test all possible paths i.e., each statement and branch is covered.

### Advantages of White Box Testing:

- Forces test developer to reason carefully about implementation.
- Reveals errors in hidden code.
- Spots the Dead Code or other issues with respect to best programming practices.

### **Disadvantages of White Box Testing**

- Expensive as one has to spend both time and money to perform white box testing.
- In-depth knowledge about the programming language is necessary to perform white box testing.

### **Integration Testing:**

The units or modules are to be integrated which gives rise to integration testing. The purpose of integration testing is to verify the functional, performance, and reliability between the modules that are integrated.

### **Integration Strategies:**

- Big-Bang Integration
- Top Down Integration
- Bottom Up Integration
- Hybrid Integration

### **Advantages:**

- Integration testing is necessary to verify whether the software modules work in unity.
- It provides a systematic technique for assembling a software system while conducting tests to uncover errors associated with interfacing.

### **Disadvantages:**

- Since all modules are coupled, if some fault arises in the systems, it is difficult to spot it on.
- Time taken by the entire software system is much more than other integration testing approaches.

## **6.4 Test Cases:**

- It's an eye detection test case where we input as counter=1 in blink detection where we execute the program but it arise an exception.
- It is an test case where we check the total number of blinks with consecutive frames = 1 blinks should be minimum but blinks number is extending the minimum.
- It is an test case where the location should mention correctly in main file or else it does not decode the code bytes and program doesn't execute.

## CONCLUSION

### **Result:**

In order to demonstrate a prototype of the approach, we have constructed a python script that takes live videos as source of inputs. We designed the script performing all the mentioned data. The script has been attached with this doc, by having a glance on the script you will come to know that the above-mentioned techniques are used. This script is designed in a way to display the real time EAR on the screen

### **Summary:**

In this chapter, we have seen the workflow of the project, its main components, their interaction, some intermediate results, brief explanation of some function. We clearly understood some concepts regarding the image processing, necessity for transition of normal image to gray scale, facial landmark predictor and how it helped in locating the eyes, constructing contours around them and visually observe the dynamic motion of eyes and track them.

## REFERENCES

- [1] N.Mathiarasi, T.Sureshkumar,"A Survey on Driver's Drowsiness and Unconsciousness Detection Methodologies",in International Journal of Engineering Development and Research,2014.
- [2] Dr. Amitabh Wahi, Prof. S. Sundaramurthy, Ms P. Abinaya,"A Survey on Automatic Drowsy Driver Detection System in Image Processing", in International Journal Of Innovative Science And Applied Engineering Research (IJISAER),2014.
- [3] Omkar Dharmadhikari,Revati Bhor,Pranjal Mahajan,H.V. Kumbhar,"Survey on Driver's Drowsiness Detection System", International Journal of Computer Applications (0975 – 8887)Volume 132 – No.5, December 2015.
- [4] Binita Sumant Singh, Ravi Krishan Pandey," A Survey on Drowsy Detection Technology" is available from [www.ijariie.com](http://www.ijariie.com).
- [5] Asad Ullah, Sameed Ahmed, Lubna Siddiqui, Nabiha Faisal." Real Time Driver's Drowsiness Detection System Based on Eye Conditions",in International Journal of Scientific & Engineering Research, Volume 6, Issue 3, March-2015.
- [6] Jay D. Fuletra, Dulari Bosamiya ,"A Survey on Driver's Drowsiness Detection Techniques" in International Journal on Recent and Innovation Trends in Computing and Communication.