Detection of a Drive Drowsiness based on Eyes closure and Yawning using Aspect Ratio

**ABSTRACT**

In the contemporary fast-paced lifestyle, individuals are often pressed for time and seek to optimize their productivity even during commuting. However, the prevalence of drowsy drivers poses a significant risk of road accidents. To address this issue, an innovative approach involves the integration of drowsiness detection technology using machine learning. This technology focuses on real-time monitoring of drivers by analyzing facial landmarks, namely the eyes, nose, and mouth, to ascertain their level of alertness.

The key methodology involves employing the HAAR Cascade algorithm for facial detection, a robust technique for identifying objects in images or video. In conjunction, a supplementary cascade file containing features specific to eyes, nose, and mouth aids in pinpointing these crucial facial landmarks. Subsequently, the aspect ratio of these landmarks is calculated and compared against predefined threshold values. If the aspect ratio surpasses the specified thresholds for any of the facial features, a drowsiness alert is triggered on the display screen.

By combining facial detection with landmark analysis through machine learning, this technology provides an initiative-taking means of preventing accidents caused by drowsy driving. The system not only enhances safety during travel but also underscores the potential of artificial intelligence in mitigating risks associated with human behavior.

**Table of Content**

[1. INTRODUCTION 8](#_Toc49527334)

[**1.1.** **System Architecture** 9](#_Toc49527335)

[2. LITERATURE SURVEY 10](#_Toc49527337)

[2.1 Survey on Driver Fatigue Detection Based on Eye Tracking 10](#_Toc49527338)

[2.2 Survey on Non-intrusive drowsiness detection by employing Support Vector Machine 11](#_Toc49527340)

[2.3 Survey on Intelligent Video-Based Drowsy Driver Detection System 11](#_Toc49527342)

[2. 4. Survey on Driver Drowsiness Detection System 12](#_Toc49527344)

[2.5. Survey on Drowsy Driver Identification Using Eye Blink detection 13](#_Toc49527346)

[3. SYSTEM REQUIREMENT SPECIFICATION 14](#_Toc49527348)

[3.1 Hardware Requirement 14](#_Toc49527349)

[3.2 Software Requirements 14](#_Toc49527350)

[4. SYSTEM ANALYSIS 15](#_Toc49527351)

[4.1 EXISTING SYSTEM 15](#_Toc49527352)

[4.2 PROPOSED SYSTEM 15](#_Toc49527353)

[4.3 FEASIBILITY STUDY 15](#_Toc49527354)

[5. SYSTEM DESIGN 17](#_Toc49527355)

[5.1 INPUT DESIGN 17](#_Toc49527356)

[5.2 OUTPUT DESIGN 17](#_Toc49527357)

[5.3 UML DIAGRAMS 18](#_Toc49527358)

[5.4. MODULES 30](#_Toc49527365)

[6. SYSTEM IMPLEMENTATION 31](#_Toc49527366)

[7. SYSTEM TESTING 45](#_Toc49527380)

[7.1. SOFTWARE TESTING STRATEGIES 45](#_Toc49527381)

[8. APPENDIX 48](#_Toc49527382)

[9. CONCLUSION 51](#_Toc49527388)

[10. REFERENCES 52](#_Toc49527389)

**List of Figures**

[Fig: 1 System Model 9](#_Toc49527467)

[Fig.2 driver tiredness detection system 10](#_Toc49527470)

[Fig.3. Non-intrusive drowsiness detection 11](#_Toc49527472)

[Fig. 4 Video-Based drowsy detection 12](#_Toc49527474)

[Fig.5. System Model of Drowsiness Detection System 12](#_Toc49527476)

[Fig.6 Drowsy Detection by Eye Blink System 13](#_Toc49527478)

[Fig.7 System Use case Diagram 18](#_Toc49527490)

[Fig.8 Project Class Diagram 19](#_Toc49527491)

[Fig.9. Project Sequence Diagram 20](#_Toc49527492)

[Fig.10 Project activity diagram 21](#_Toc49527493)

[Fig.11 System DFD diagram 22](#_Toc49527494)

[Fig.12 Project Flow Chart diagram 23](#_Toc49527495)

[Fig.13 Python Setup installation 31](#_Toc49527498)

[Fig.14 Select Python features 32](#_Toc49527499)

[Fig.15 Set python installing location on local system 32](#_Toc49527500)

[Fig.16 Python Setup Installing Process 32](#_Toc49527501)

[Fig.17 PyCharm IDE installation process 33](#_Toc49527502)

[Fig.18 Choose Start Menu Folder 33](#_Toc49527503)

[Fig.19 Run PyCharm default Edition 34](#_Toc49527504)

[Fig.20 PyCharm IDE Launching 34](#_Toc49527505)

[Fig.21 Qt Designer 35](#_Toc49527506)

[Fig.22 Convert .UI files 36](#_Toc49527507)

[Fig.23 New Project Creation 36](#_Toc49527508)

[Fig.24 PyCharm IDE 37](#_Toc49527509)

[Fig.25 MySQL database 37](#_Toc49527510)

[Fig.26 System Main Frame 48](#_Toc49527514)

[Fig.27 detecting driver facial landmarks 48](#_Toc49527515)

[Fig.28 detecting driver mouth opening for yawning test 49](#_Toc49527516)

[Fig.29 detecting driver eye closing testing 49](#_Toc49527517)

[Fig.30 detect drowsy with head too low 50](#_Toc49527518)

**1. INTRODUCTION**

In the realm of machine learning, there exist user-friendly and efficient libraries that cater to the needs of artificial intelligence applications. These libraries enable the straightforward implementation of methods to meet specific requirements. In the context of our system, the primary objective is to detect driver drowsiness without the incorporation of additional sensors. The rationale behind avoiding sensor integration lies in the associated drawbacks, such as increased system costs and size. While vehicle and physiological-based technologies do offer drowsiness detection capabilities, they often come at a higher expense and entail larger system dimensions, making them less practical.

Our approach adopts a visual behavior-centric strategy, leveraging a cost-effective solution by exclusively utilizing a web camera. By capturing input images from the camera, the system initiates the process by employing the HAAR cascade algorithm for face detection and identifying facial landmarks. Subsequently, visual behavior analysis, including eye aspect ratio, mouth open aspect ratio, and nose length aspect ratio computations, is performed using predefined threshold values.

The system scrutinizes specific visual behaviors, such as yawning (threshold greater than 0.6), head bending (threshold between 0.7 and 1.2), and eye aspect ratio exceeding the defined limit. If these conditions align, a drowsy alert is generated. Additionally, if the mouth and nose ratios do not meet the criteria but the eye ratio satisfies the threshold, a drowsy alert is triggered to ensure the driver remains attentive.

By relying solely on visual behaviors and eschewing the need for additional sensors, our system not only effectively detects driver drowsiness but also minimizes costs and system size, making it a practical and economical solution for enhancing road safety.

* 1. **System Architecture**

A diagram of a car sharing system

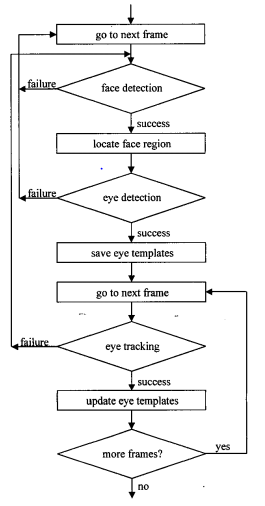
Description automatically generated

# Fig: 1 System Model

Illustrated in Figure 1 is the proposed model for our system. The initial step involves activating the web camera, which captures frames for a brief duration. These frames are then processed to generate an image, upon which facial detection is performed. The system identifies crucial facial landmarks such as eyes, mouth, and head using the extracted frames. Subsequently, ratios for these landmarks are computed, and the obtained values are compared against predefined threshold values. If any of these thresholds are exceeded, a drowsy alert is triggered to alert the driver and prevent them from falling asleep.

**2. LITERATURE SURVEY**

## 2.1 Survey on Driver Fatigue Detection Based on Eye Tracking

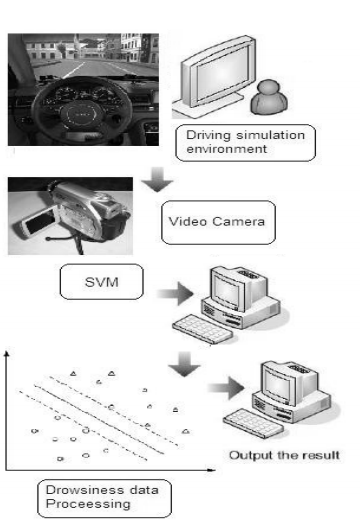
****

# Fig.2 driver tiredness detection system

C. Y. Chen et al. proposed a driver tiredness detection system utilizing eye tracking. The methodology involves initially capturing facial images using a color camera to identify the skin color. The images are then converted into grayscale to eliminate color information, leaving a black and white representation. Subsequently, the Canny edge detection algorithm is applied to unveil the contours of the eyes.

Once the eyes are identified, the system calculates the white points on the edges for every twenty frames. If a pattern indicative of driver tiredness is detected through this analysis, an alert mechanism is triggered. This alert typically takes the form of a sound alarm, serving as a timely warning to the driver to ensure safe and vigilant driving practices, thereby mitigating the risk of potential harm.

## 2.2 Survey on Non-intrusive drowsiness detection by employing Support Vector Machine

****

# Fig.3. Non-intrusive drowsiness detection

*J. Mellor et al*. proposed a non-intrusive drowsiness detection system using Support Vector Machine (SVM) classification. The approach involves utilizing a training dataset comprising attributes such as nose, mouth, and eye static values, along with a target column indicating drowsiness status (yes or no). The system initiates by activating a color camera, capturing images, and continuously identifying facial landmarks within those images. Subsequently, three threshold values are obtained, and the system predicts the drowsiness status based on the extracted features.

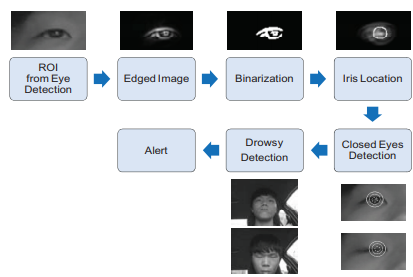
This Non-intrusive drowsiness detection method ensures real-time monitoring without intrusive measures. By employing SVM classification and leveraging facial landmarks, the system effectively evaluates the drowsiness status of the driver. If a potential drowsiness scenario is predicted, the system issues an alert, contributing to initiative-taking measures for safer driving practices.

## 

## 2.3 Survey on Intelligent Video-Based Drowsy Driver Detection System

*C. W. Chang et al*. introduced an innovative video frame-based method for detecting drowsy drivers, specifically addressing the challenge of identifying facial landmarks even when the driver is wearing spectacles. In contrast to earlier systems that struggled with accurately identifying facial landmarks when glasses were worn, this method achieves a remarkable precision of 95%, irrespective of whether the driver is wearing glasses or not.

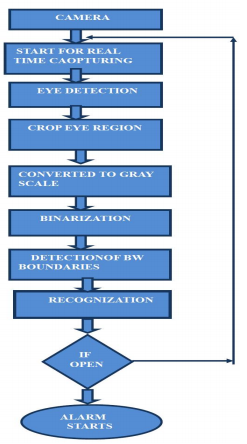
This advancement ensures a more reliable drowsiness detection system, significantly improving the system's capability to accurately assess a driver's alertness. By effectively addressing the challenges associated with facial landmarks obscured by spectacles, the method enhances the overall accuracy of identifying drowsiness, contributing to safer driving practices.

****

# Fig. 4 Video-Based drowsy detection

## 

## 2. 4. Survey on Driver Drowsiness Detection System

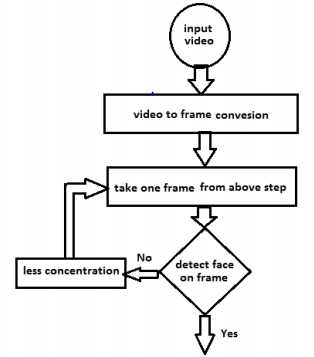
****

# Fig.5. System Model of Drowsiness Detection System

The authors *A. S. Baquhaizel et al.* were implemented detection of driver drowsy system. Here this report depicts how to detect the eyes, and also how to identify if the eyes are open or closed. Once the face landmark is located, the eyes are found by computing the horizontal averages in the face landmark i.e. Once the eyes are obtained, calculating the distances between the potency modifications in the eye landmark identify whether the eyes are opened or closed then the system conclude that the driver is get tidiness and alert a warning alarm to driver.

## 

## 2.5. Survey on Drowsy Driver Identification Using Eye Blink detection.

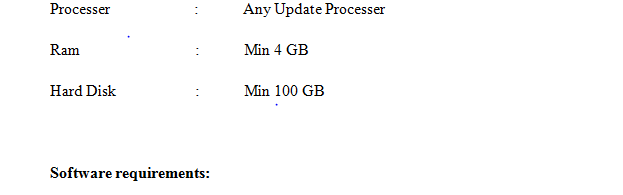
****

# Fig.6 Drowsy Detection by Eye Blink System

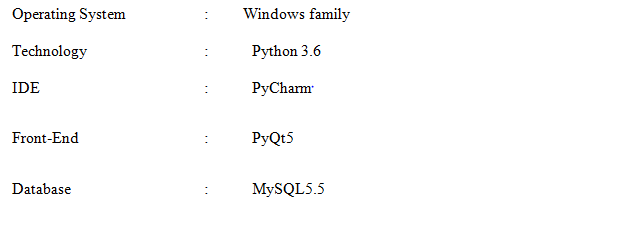
*J. N. Borole et al.* were developed eye blink drowsy detection. In this detection system it can get images from camera and with help of cascade network model file this system can detect face landmarks. While driver driving the vehicle first it located the eyes and starts to count eye blinking time, compare with mentioned adaptive threshold value if it is outrivaled then it is preventing accidents by generate alert message to driver for not to sleeping.

**3. SYSTEM REQUIREMENT SPECIFICATION**

## 3.1 Hardware Requirement



## 3.2 Software Requirements



# 4. SYSTEM ANALYSIS

## 4.1 EXISTING SYSTEM

In previous methodologies like vehicle based and physiological based are used for detecting driver drowsiness but using of physiological based like ECG, Heartbeat sensors are attached on driver but detecting fatigue level, in this case depends on sensors are used in this technique system cost and size will increase. Due to this disadvantages we had to developed low-cost level technique which is need not use any sensors for detecting driver drowsiness, by using of visual behavior with machine learning techniques.

## 4.2 PROPOSED SYSTEM

The suggest system which is visual behavior based for detecting driver drowsiness with help of eye aspect ratio, mouth opening ratio and nose length ratio are computed and depending on their values, drowsiness is detected based on developed thresholding. It takes input as image which is captured from web camera and apply HAAR cascade algorithm for detecting face and finding landmarks of eyes, mouth and nose, if any ratio satisfy then system can generate alert for making driver lose the drowsiness.

## 4.3 FEASIBILITY STUDY

A feasibility study is used to Fig out usable of an idea, which is making sure the project is technically and economically feasible as well as justifiable. The requiring too many resources which leads to restrict the other resources performing on the tasks and also the company would not get profit by taking the project back. A feasibility study estimates the company project’s quality for success. There are three types of feasibility study which is described below:

**4.3.1Technical Feasibility**

In this feasibility it can determine the technical resource availability to company. The feasibility can useful for companies to estimate the technical resources and team has capacity to convert the theoretical to working system. It is also evaluates the others technical resources, software and hardware of the present system.

**4.3.2 Economical Feasibility**

The Economical Feasibility involves in cost analysis of the project, which is helping company decide feasible, cost and profits affiliated with projects before financial resources are distributed. It delivers an independent project evaluation estimate project reliability helping to decision makers recognize the good economic profits to the company that the suggest project could be delivered.

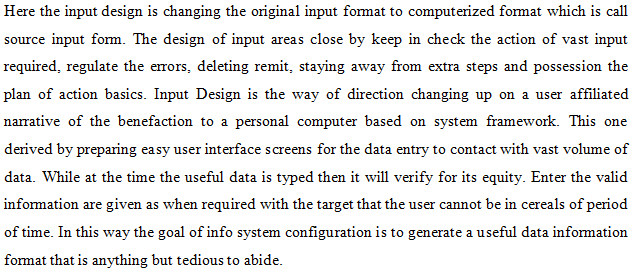
**4.3.3. Social Feasibility**

This is a one of the feasibility study where the welcoming of the people is taking into account pay attention to the product to be starting the ball rolling. This feasibility describes how the folks propose to determine user co-operation before changes are introduced. As well as it characterize the effect on folks from the basic explanation of the new system observing whether there will be a need for teach new skills for the workforce.

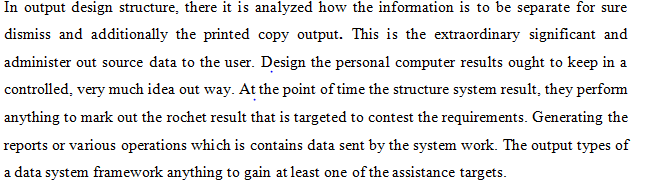
# 5. SYSTEM DESIGN

The system design is the procedure of explain system modules, interfaces and architecture to satisfy the system requirements.

## 5.1 INPUT DESIGN

****

## 5.2 OUTPUT DESIGN

****

## 5.3 UML DIAGRAMS

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

**5.3.1 USE CASE DIAGRAM:**

****

# Fig.7 System Use case Diagram

**Description:**

From the above use case diagram it explain about what user can perform use cases that means he/she can run the webcam and read the image continuously from that image face landmarks will be detect and compare with adaptive threshold values and detecting driver drowsiness.

**5.3.2. CLASS DIAGRAM:**

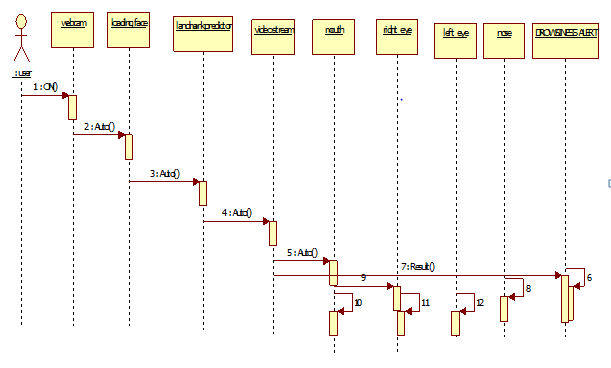


# Fig.8 Project Class Diagram

**Explanation:**

From the above class diagram it can reveal that it contains webcam and Video Stream which is has predefined library for capturing image and drowsiness for detecting driver drowsiness.

**5.3.3. SEQUENCE DIAGRAM**



# Fig.9. Project Sequence Diagram

**Description:**

From above sequence diagram user can now the flow of execution of system and first what user have to do after start the web camera and detecting facial land marks and detecting drowsy.

**5.3.4 ACTIVITY DIAGRAM:**

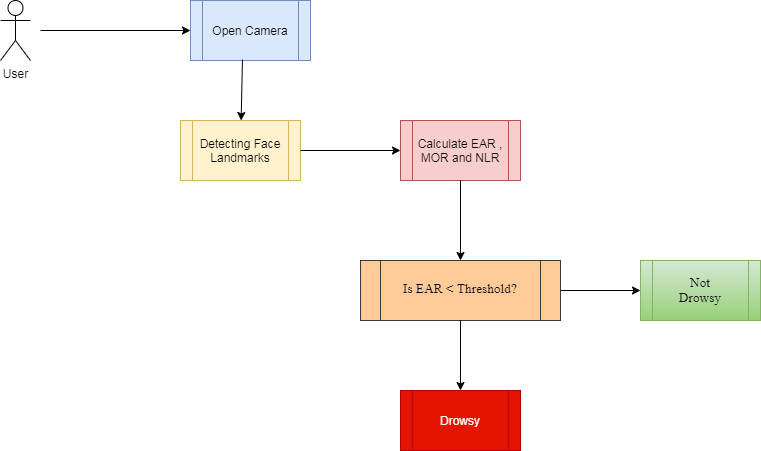


# Fig.10 Project activity diagram

**Description:**

From the above diagram it tells about activity at what state user have to be apply and perform the application calculation and check validation of application if valid user then only they can allow accessing the system.

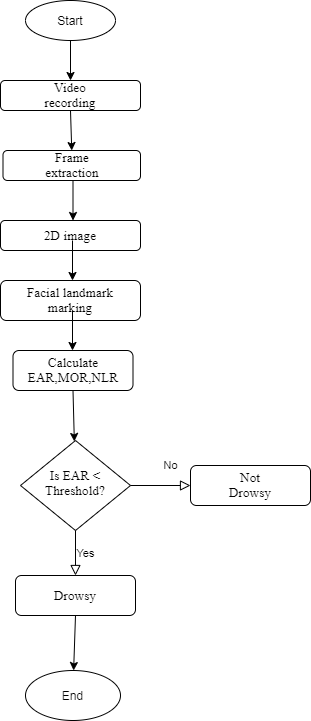
**5.3.5 .DATA FLOW DIAGRAM:**

****

# Fig.11 System DFD diagram

In this system DFD diagram describes data flow between two states. Here user starts with the state image acquiring and later it can connect to another state like that it can flow from creation to termination.

**5.3.6 FLOW CHART DIAGRAM**

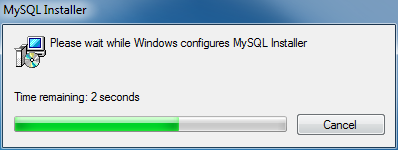
****

# Fig.12 Project Flow Chart diagram

In this system flow chart diagram shows about how the system functionality starts from and perform the modules if validations satisfy until approach the ending functionality.

**5.3.7. Data Base Tables**

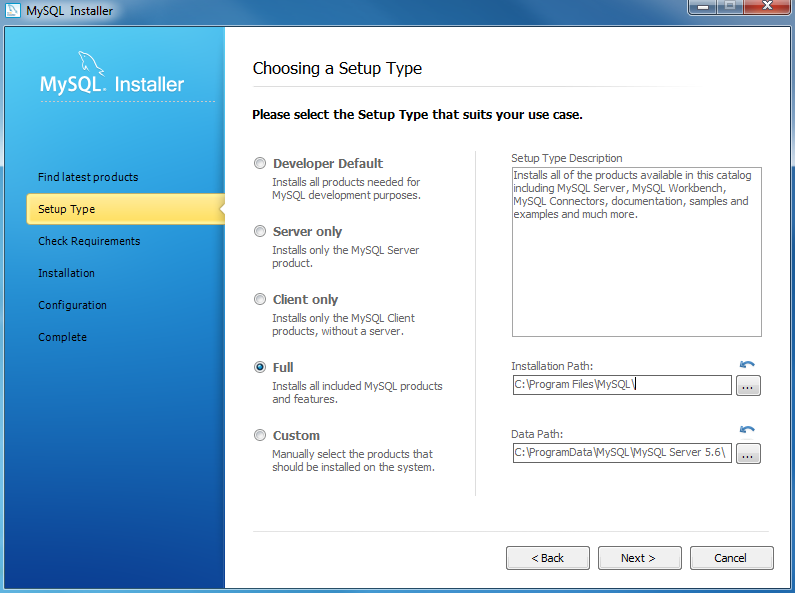
Download MySQL Installer from Office website <http://dev.mysql.com/downloads/installer/>.



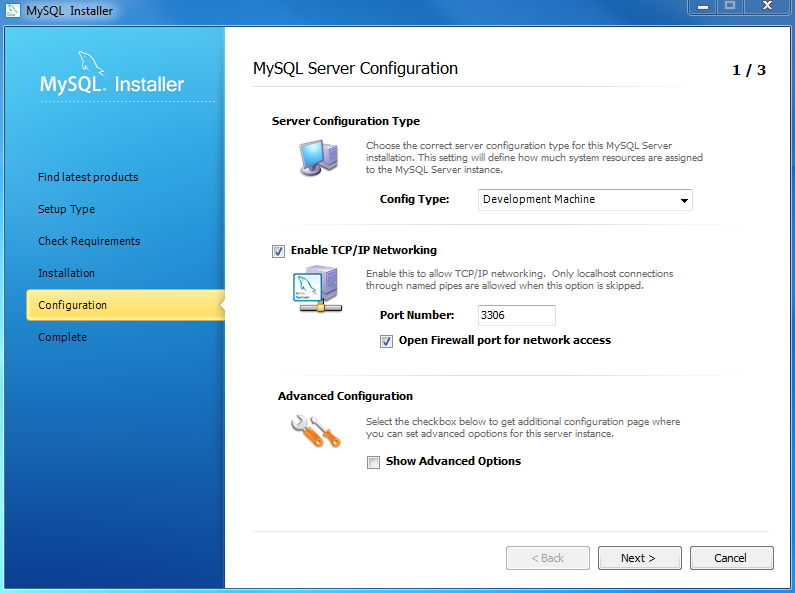
Install MySQL Step 1: Windows con Figs MySQL Installer



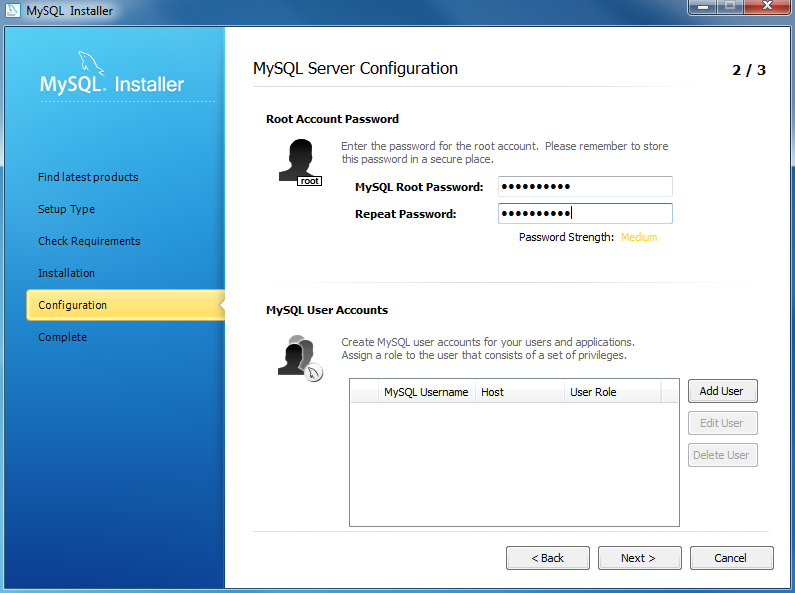
Select Install MySQL Products



Select Full version and disk space for installing

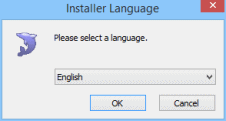


Set the connection port number “3306”

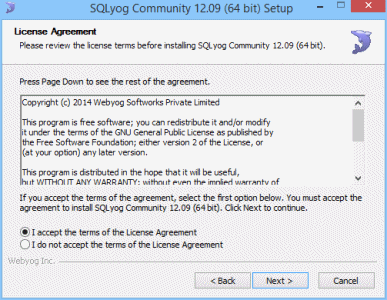


Set Password & Conform password

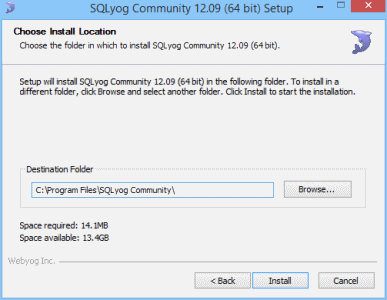
**Install SQLyog community edition**



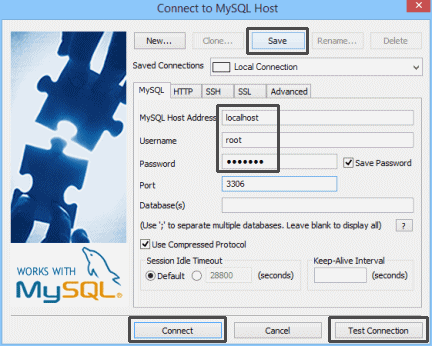
Select the Language



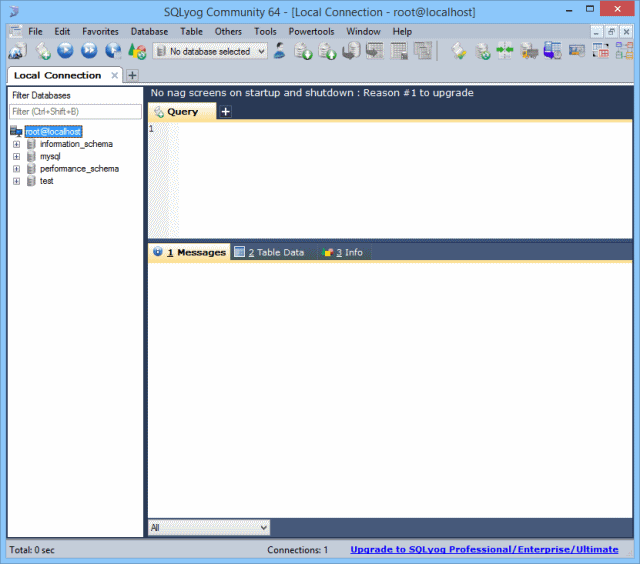
Accept terms of the software



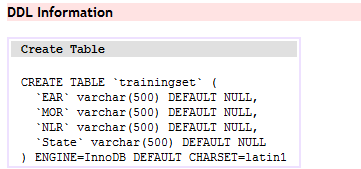
Selecting the installing location in our system



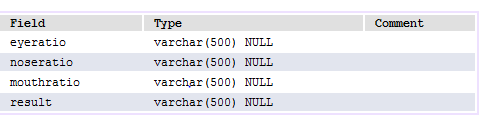
Create a New MYSQL Connection



Create any New Database

****

**Table Name:** dataset

****

## 5.4. MODULES

**Data Acquisition:**

In this module the frames are capturing from web camera and convert these frames into 2D image. While running the camera the user staring at 30 seconds for detecting face landmarks.

**Face Detection:**

In this module, after taking input as image from camera then it can detect the face coordinates with help of Haar cascade file. With help of dlib python library it can load that cascade file and applying the get\_frontal\_face\_detector () for face detection.

**Facial Landmark:**

The completion of face detection with help of shape predictor network model file can be loaded by dlib library. This method can take the input image as face detection image and invoke the predictor method for detecting face land marking.

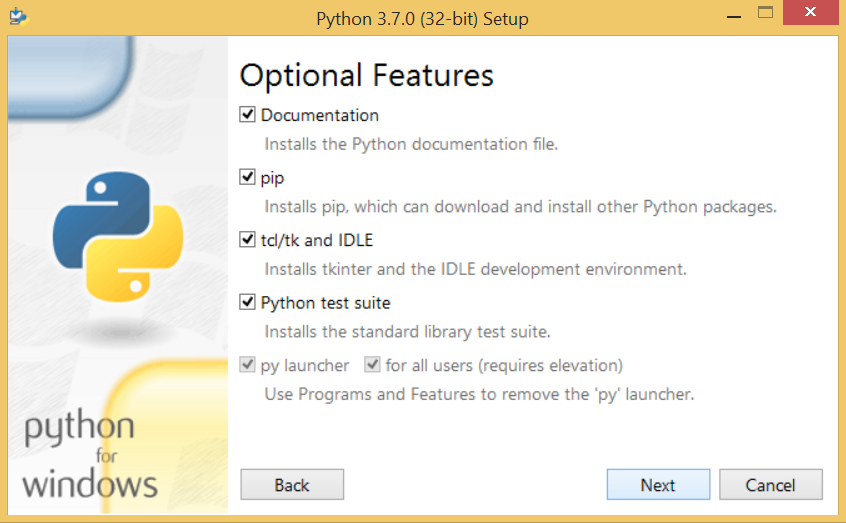
**Drowsy Detection:**

Here after detection facial landmarks it starts to calculate the three ratios like EAR, MOR and NLR with their applying Euclidian distance formulas. At every stages the three ratios threshold values calculate and compare with existing threshold values if any conditions salsify then generate drowsy alerts by eye closing, head bending and yawning.

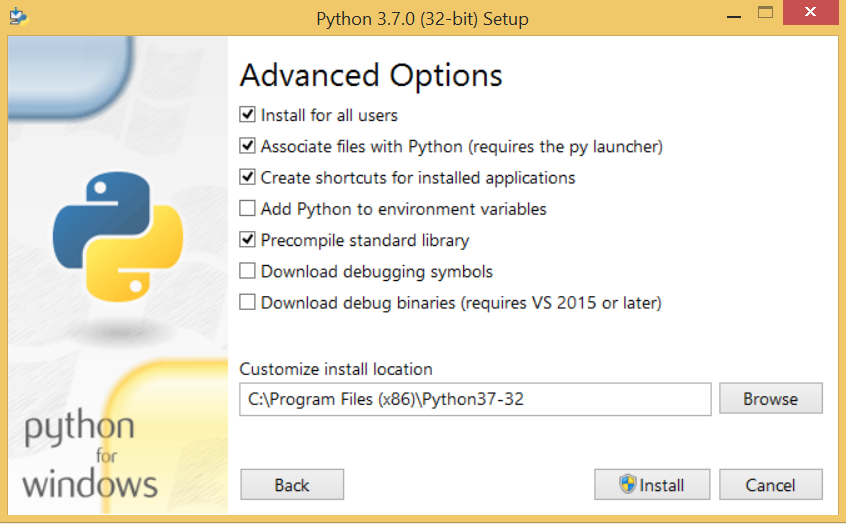
# 6. SYSTEM IMPLEMENTATION

****

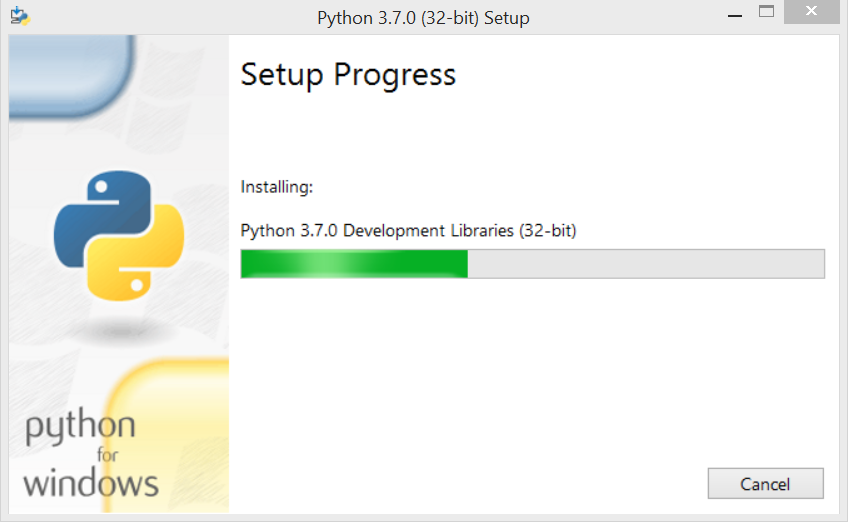
# Fig.13 Python Setup installation

For installing python software in your laptop we need to install this setup file and check the Add python to Path for keep the default python environment.

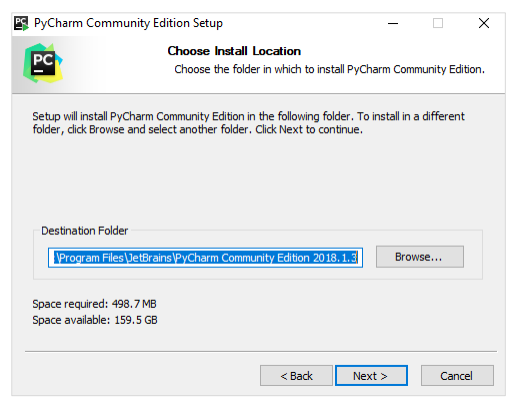
# Fig.14 Select Python features



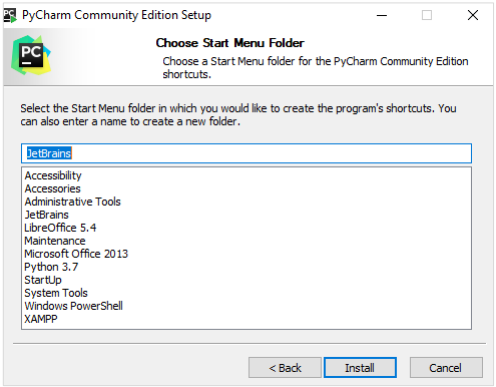
# Fig.15 Set python installing location on local system



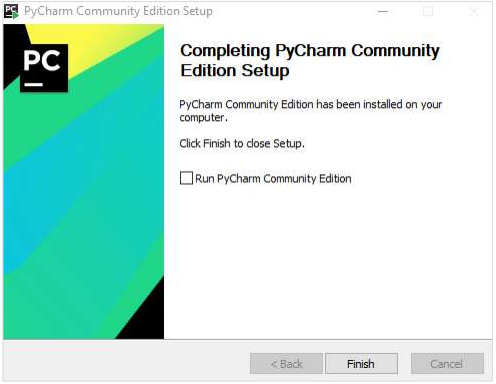
# Fig.16 Python Setup Installing Process

****

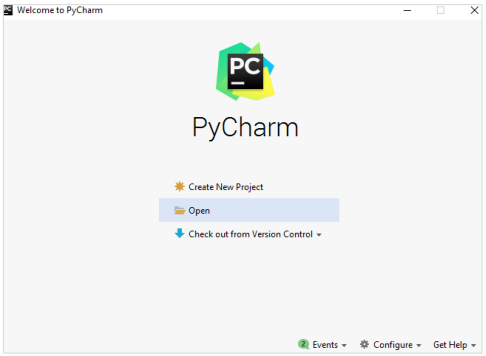
# Fig.17 PyCharm IDE installation process

****

# Fig.18 Choose Start Menu Folder

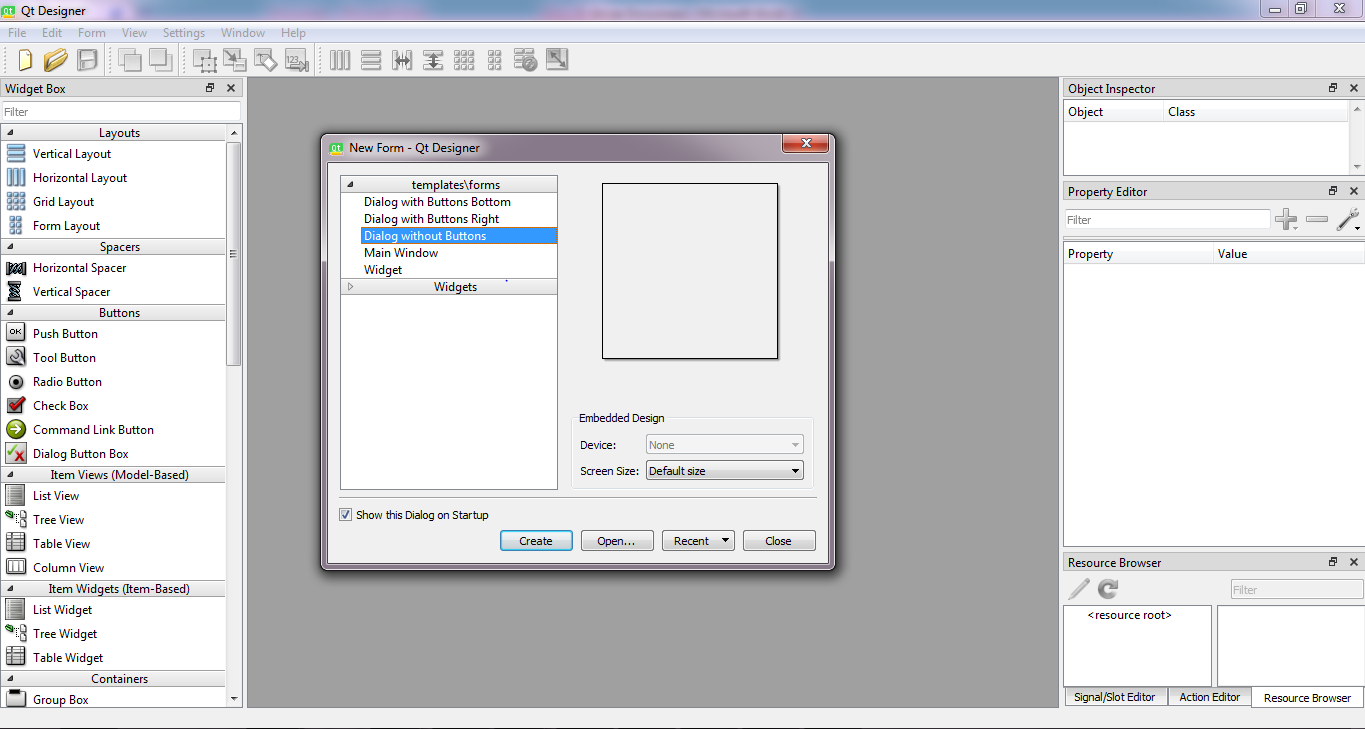


# Fig.19 Run PyCharm default Edition



# Fig.20 PyCharm IDE Launching

**PyQt5:**

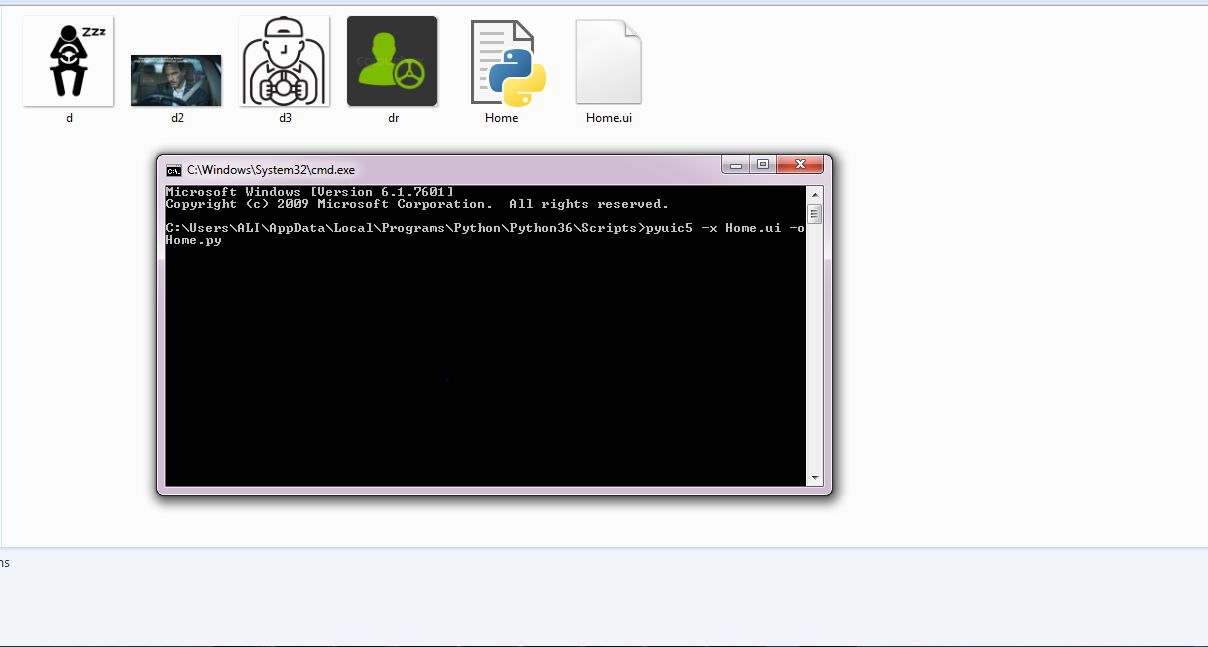
****

# Fig.21 Qt Designer

In python language by installing pyqt5-tools we can get Qt Designer application. The Fig.13 depicts the Qt Designer proto type. Using Qt Designer we can design the front end applications for user interface. This tool generates ‘.UI’ format files.

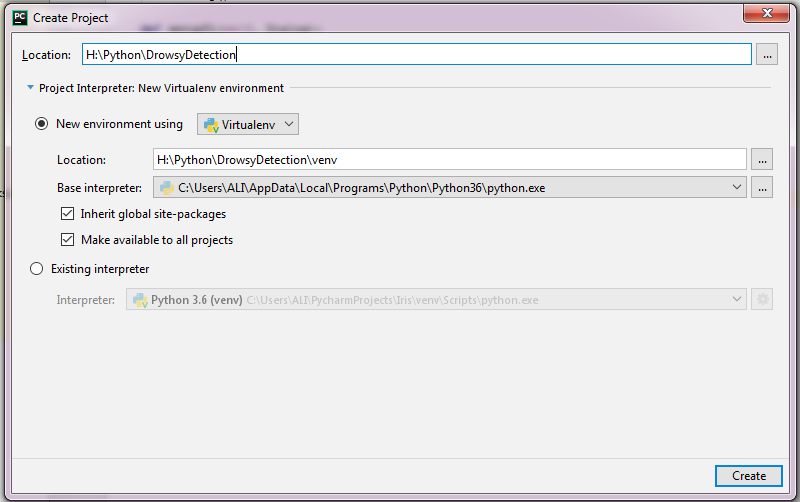
**Convert UI files to Python files:**

Fig.14 shows about ‘.UI’ conversions files commands, by using pyuic5 tools we can convert from .UI to .PY files.

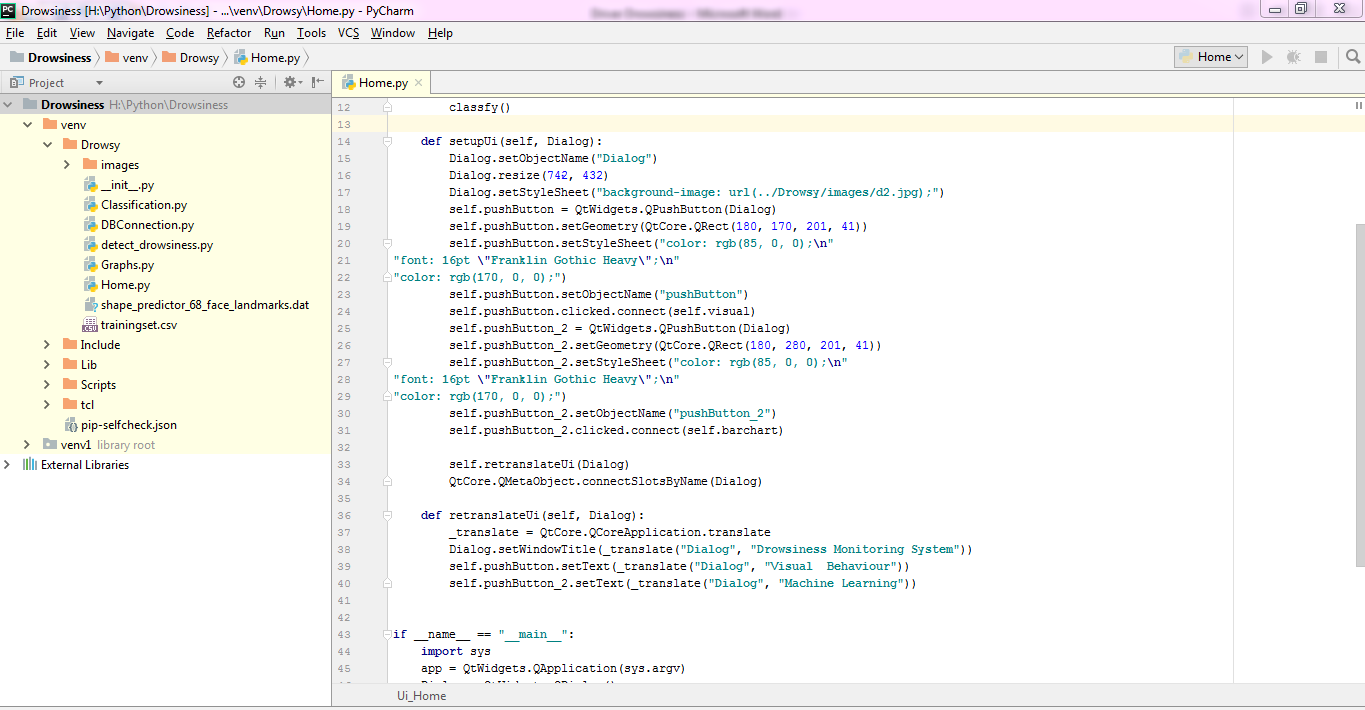


# Fig.22 Convert .UI files

**PyCharm IDE:**

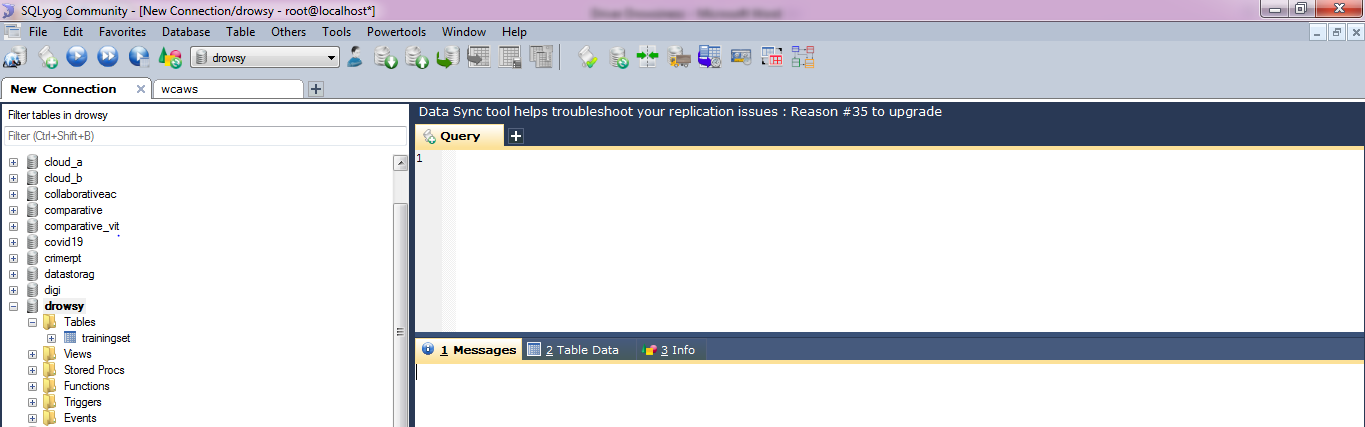
****

# Fig.23 New Project Creation

****

# Fig.24 PyCharm IDE

Fig 15 & 16 depicts about how to create new project in PyCharm IDE and deploy the python code in IDE. In this PyCharm IDE, the python code will be developed for achieve our system implementation.

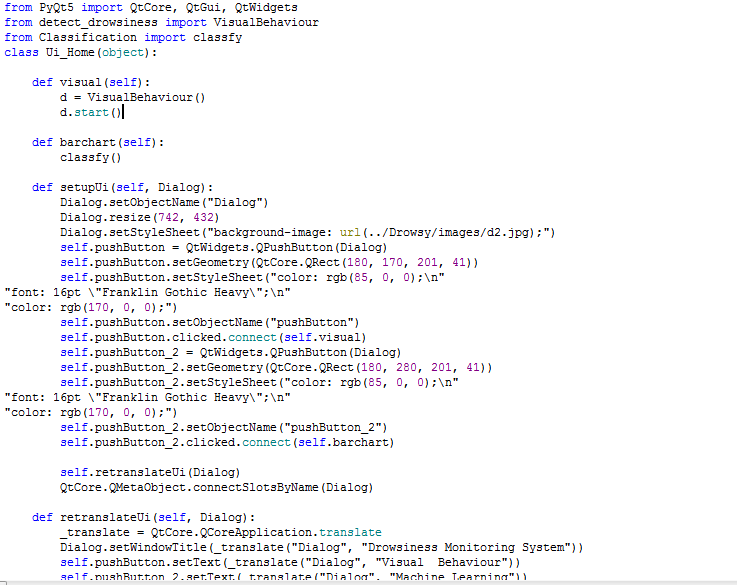
****

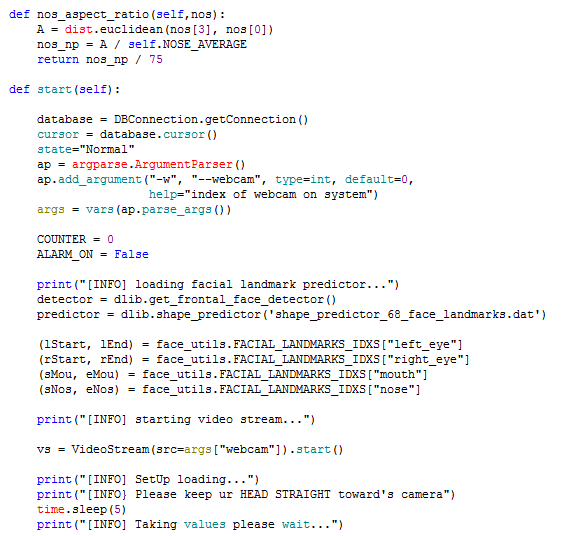
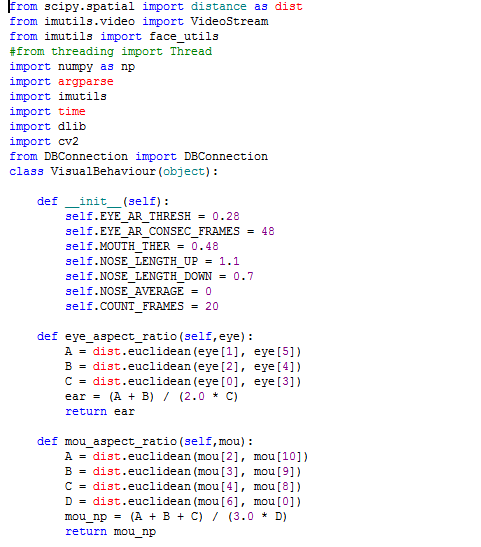
# Fig.25 MySQL database

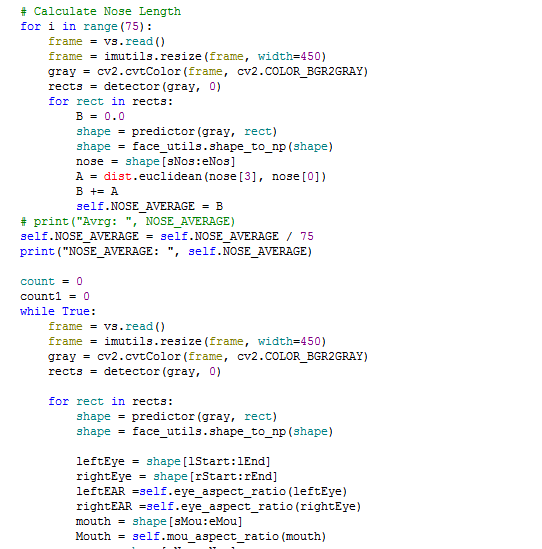
For storage purpose we install mysql-coonector software in our system. Create new data base with name of drowsy database and it contains training set table.

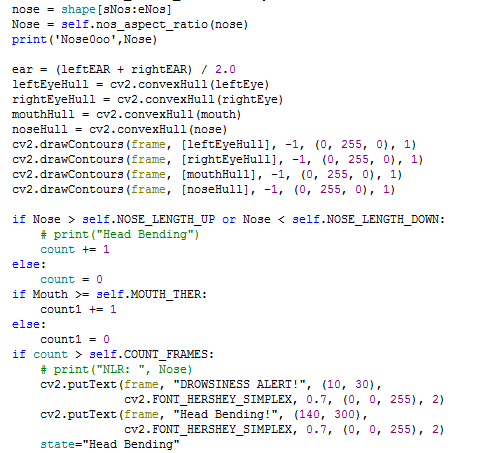
**6.1. SAMPLE CODE:**

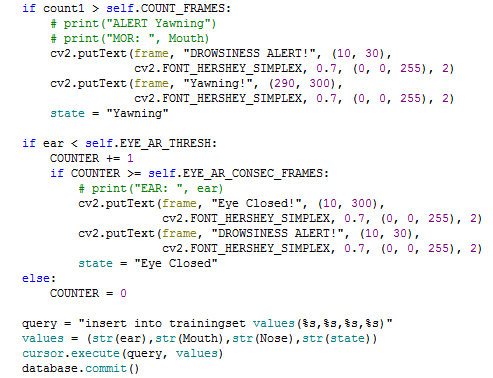
**Home.py:**

****

**detect\_drowsiness.py**

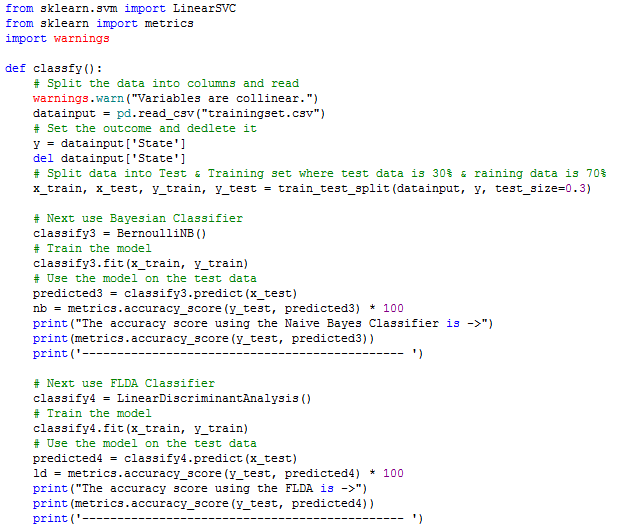


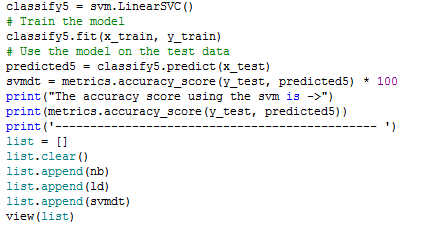




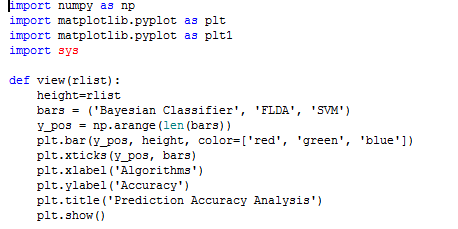
**DBConnection.py:**

 **Classification.py:**





**Graphs.py:**



# 7. SYSTEM TESTING

## 7.1. SOFTWARE TESTING STRATEGIES

Once completion of application development software testing is perform important role. While testing the application our system follow the some testing strategies like unit testing, integrating testing, functional testing, system testing etc.

**7.1.1 System Unit testing**

Here the software application can be divide into small pieces modules like register, login. The developer will write the code for that forms input presentation logic and evaluating that small unit of module working properly or not.

**7.1.2 Project Integration testing**

In the integration testing, the sample code will be tested for calculate system application performance. As well as it will evaluate the application integrated into suitable for different integrated development environment tools.

**7.1.3 Application Functional test**

The functional testing useful for the testing validation in particular forms. Whether the input fields are working properly with validations like length validation, input fields are required fields etc.

**7.1.4. Application System Test**

In system testing it will check whether the application will work on low level and high level to machine language and it can check portable or not, more on it will platform independent or dependable core system.

**7.1.5. System White Box Testing**

This testing is the testing of a software infrastructure and internal sources. As well as this testing can checking for security and input, output flow application and user friendly designing improvement.

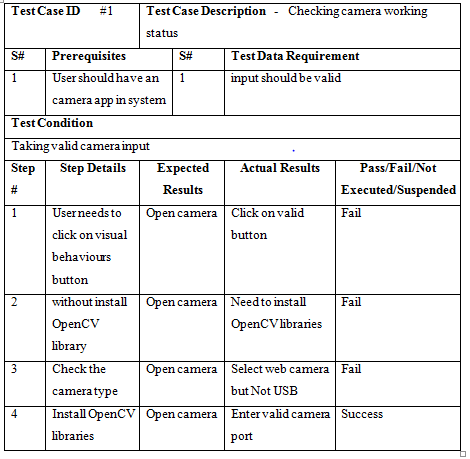
**7.1.6. System Black Box Testing**

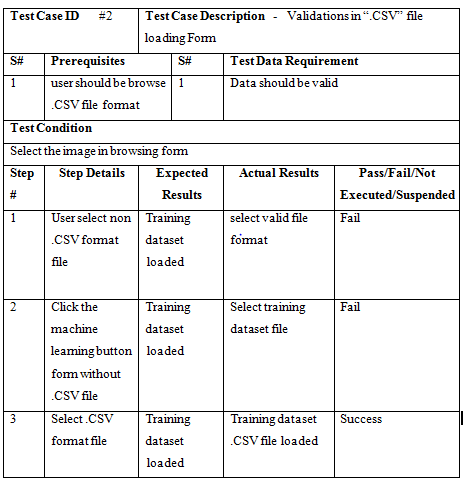
This testing can focus on input and output of software system without bothering and lack of project knowledge information. The test cases working on software specifications and requirements.

**7.1.7. System Acceptance Testing**

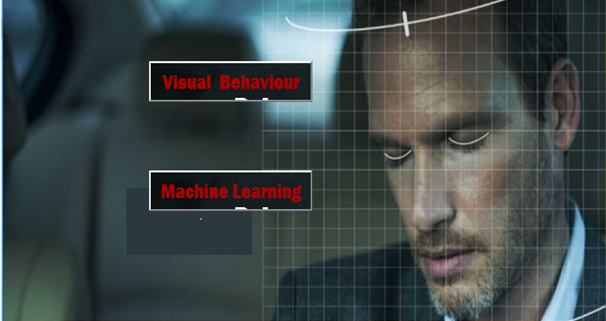
This testing useful to the system testing that is performed by the customer to determine the product accept the delivery of the system.

**TEST CASES**

****

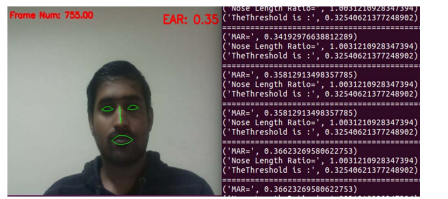


# 8. APPENDIX



# Fig.26 System Main Frame

Here when you run the Main.py file then this frame will be open with two buttons and background image.

****

# Fig.27 detecting driver facial landmarks

Here with help of shape predictor model file the system can detect facial landmarks.

A person with his mouth open

Description automatically generatedA person with green eyes and text overlay

Description automatically generated

# Fig.28 detecting driver mouth opening for yawning test

From the above picture it can detect drowsy alert by opening mouth by calculate the MOR distance.

**A person with a beard and mustache

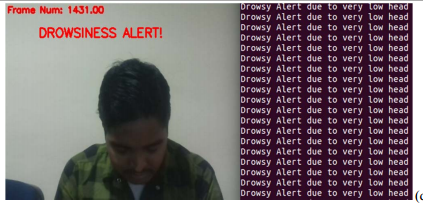
Description automatically generated**A person with green eyes and text overlay

Description automatically generated

# Fig.29 detecting driver eye closing testing

From the above picture it can detect drowsy alert by close the eyes by calculate the EAR distance.

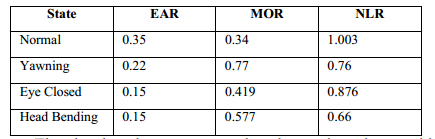
A person with a beard and mustache

Description automatically generated

# Fig.30 detect drowsy with head too low

From the above picture it can detect drowsy alert by head bending by calculate the NLR distance.

Table I: Training dataset



# 9. CONCLUSION

In this system we can get accurately detect driver drowsiness with help of three aspect ratios which is getting from visual behaviors method as well as we are implemented SVM classifier for predicting drowsy status like normal, yawing, head bending, eye closing with help of training dataset, this is also get 95% accuracy with compare naïve bayes classifier. So that previous methods like physical or vehicle methods are giving very less detecting driver drowsy results if it is compare with current system.

# 10. REFERENCES

[1] W. L. Ou, M. H. Shih, C. W. Chang, X. H. Yu, C. P. Fan, "Intelligent Video-Based Drowsy Driver Detection System under Various Illuminations and Embedded Software Implementation", 2015international Conf. on Consumer Electronics - Taiwan, 2015.

[2] W. B. Horng, C. Y. Chen, Y. Chang, C. H. Fan, “Driver Fatigue Detectionbased on Eye Tracking and Dynamic Template Matching”, IEEEInternational Conference on Networking,, Sensing and Control, Taipei,Taiwan, March 21-23, 2004.

[3] S. Singh, N. P. papanikolopoulos, “Monitoring Driver Fatigue usingFacial Analysis Techniques”, IEEE Conference on IntelligentTransportation System, pp 314-318.

[4] B. Alshaqaqi, A. S. Baquhaizel, M. E. A. Ouis, M. Bouumehed, A.Ouamri, M. Keche, “Driver Drowsiness Detection System”, IEEEInternational Workshop on Systems, Signal Processing and theirApplications, 2013.

[5] M. Karchani, A. Mazloumi, G. N. Saraji, A. Nahvi, K. S. Haghighi, B. M.Abadi, A. R. Foroshani, A. Niknezhad, “The Steps of ProposedDrowsiness Detection System Design based on Image Processing inSimulator Driving”, International Research Journal of Applied and BasicSciences, vol. 9(6), pp 878-887, 2015.

[6] R. Ahmad, and J. N. Borole, “Drowsy Driver Identification Using Eye Blink Detection,” IJISET - International Journal of Computer Science and Information Technologies, vol. 6, no. 1, pp. 270-274, Jan. 2015.

[7] A. Abas, J. Mellor, and X. Chen, “Non-intrusive drowsiness detection by employing Support Vector Machine,” 2014 20th International Conference on Automation and Computing (ICAC), Bedfordshire, UK, 2014, pp. 188- 193.

[8] A. Sengupta, A. Dasgupta, A. Chaudhuri, A. George, A. Routray, R. Guha; "A Multimodal System for Assessing Alertness Levels Due to Cognitive Loading", IEEE Trans. on Neural Systems and Rehabilitation Engg., vol. 25 (7), pp 1037-1046, 2017.

[9] K. T. Chui, K. F. Tsang, H. R. Chi, B. W. K. Ling, and C. K. Wu, “An accurate ECG based transportation safety drowsiness detection scheme,” IEEE Transactions on Industrial Informatics, vol. 12, no. 4, pp. 1438-1452, Aug. 2016.

[10] N. Dalal and B. Triggs, “Histograms of Oriented Gradients for Human Detection”, IEEE conf. on CVPR, 2005.

[11] V. Kazemi and J. Sullivan; "One millisecond face alignment with an ensemble of regression trees", IEEE Conf. on Computer Vision and Pattern Recognition, 23-28 June, 2014, Columbus, OH, USA.

[12] Richard O. Duda, Peter E. Hart, David G. Stork, “Pattern Classification”, Wiley student edition.