SOFTWARE DESIGN SPECIFICATION

1.0 Introduction

This section provides an overview of the entire design document. This document describes all data, architectural, interface and component-level design for the software.

Real time surveillance using object detection is software which will be used to detect the driver's activity while the object is driving the car. It catches the activities that may cause an accident and can be helpful to alert the driver to be cautious about the situations on the road. This document provides the information about the various modules that are implemented in this project such as

The product is to build python-based software which shows fatigue in the driver that functions in a way where the opening and closing of eyes or blink rate figure out whether the driver is drowsy. Another factor which is the yawn rate figures out if the individual is drowsy and when these features are captured an alarm is set off to alert the driver to prevent fatal accidents

1.1 Goals and objectives

Overall goals and software objectives are described.

According to FMCA (Federal Motor Carrier Safety Administration) in the USA it is said that Fatigue is the major factor to cause accidents where fatigue shows about 64% of truck drivers experience fatigue regularly. Earlier studies indicated that drowsy driving could be involved in upwards of 40% of truck crashes and about 50% of accidents involving driver fatigue take place between midnight and 8 am. By applying our computer vision algorithm with the help of OpenCV we can clearly detect whether the person is drowsy or awake. If the driver is fatigued, then an alarm is set off to wake the driver up.

1.2 Statement of scope

A description of the software is presented. Major inputs, processing functionality, and outputs are described without regard to implementation detail.

As mentioned above the software along with the camera will be installed in a car in such a way that it only captures the driver of the car which will be linked to the laptop. The driver's action such as blinking of eyes and yawning will be

1.3 Software context

The software is placed in a business or product line context. Strategic issues relevant to context are discussed. The intent is for the reader to understand the 'big picture'.

The user wants a system that could be used to monitor the moment of driver in case of drowsiness and fatigue to prevent accidents.

1.4 Major constraints

Any business or product line constraints that will impact the way the software is to be specified, designed, implemented, or tested are noted here. Constraints include an easy-to-use interface, and web-based platform or, at bare minimum, a Mac/PC based local system. The user wants a system that could be used to monitor the moment of driver in case of drowsiness and fatigue to prevent accidents.

2.0 Data design

Data design is a very important aspect of the project which helps us to run the project faster and more efficiently. It defines how the data will be used in the project or how the functionality of the project is dependent on the data structures that are being used while implementing the project.

2.1 Data structures

The Data structures that are used in this project are arrays and lists.

2.2 Database description

Database(s) created as part of the application is(are) described.

Items shall be stored on the laptop machine. Extremely high criticality. Limited network / wi-fi availability could present a technical challenge. The above stated factor is a risk we have met. Drop it by reducing the dependency of our program on these things. This requirement is the basis of the project; all other aspects depend on it.

2. The items shall be accessible via web view.

Users of the software should be able to run the application on web browsers. Extremely high criticality. We do not foresee any technical issues preventing the implementation of this. This requirement can be satisfied. This requirement depends on requirement number one.

3. The data stored should be accessible.

Items and other data should be able to be accessed. Extremely high criticality. We do not foresee any technical risks involved in this requirement. The only factor we can meet here is the user of the system not being able to use it correctly. We will overcome this by training those who will be using it. This requirement is dependent on requirements.

3.0 Architectural and component-level design requirement

Component-level design gives functionality and purpose to each component of our software by defining its interface, algorithms, data structure, and communication methods. Each input on this software branches out to the detection of drowsiness in the user and alert if the user is drowsy.

Architecture Requirements;

•The dashcam/webcam will be installed in the car/bus/cargo trailers (In the current scenario we do not have the resources to implement our software onto a vehicle but with given time it can be accomplished).

•It will be connected to the computer which will process the image of the object which is driving the car.

•The software will process real-time video and identify various facial details to detect the EAR and MAR of the user.

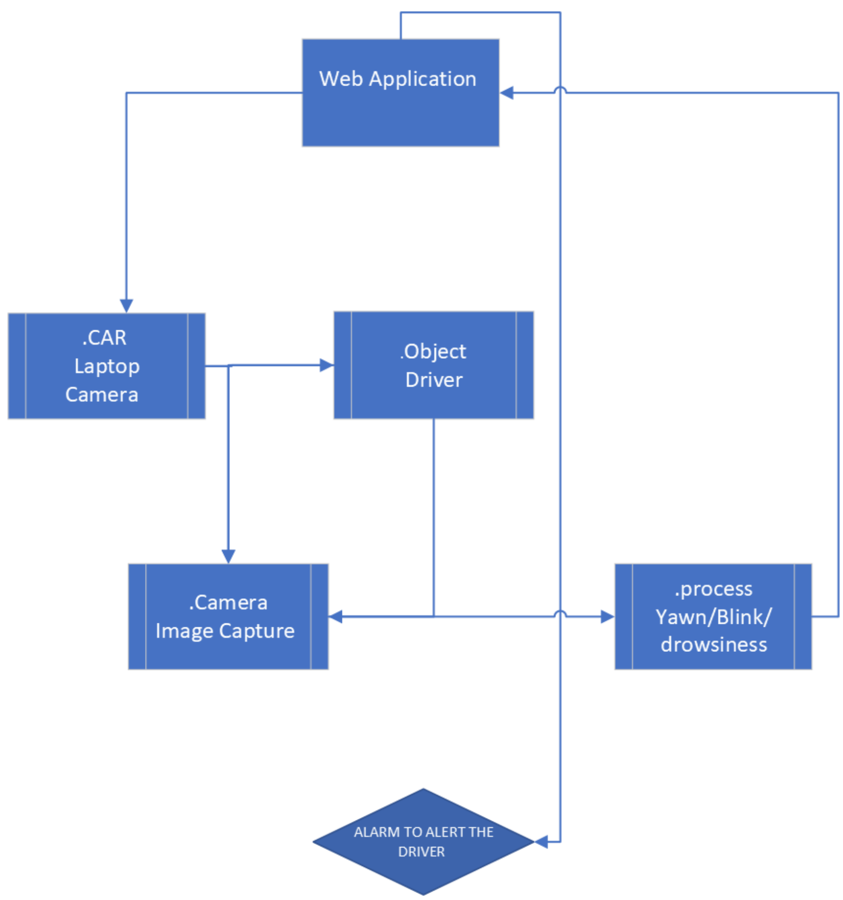
•After processing the real-time video if it feels there is something to warn the driver about it will alert the driver with an alarm.

3.1 Architecture diagrams

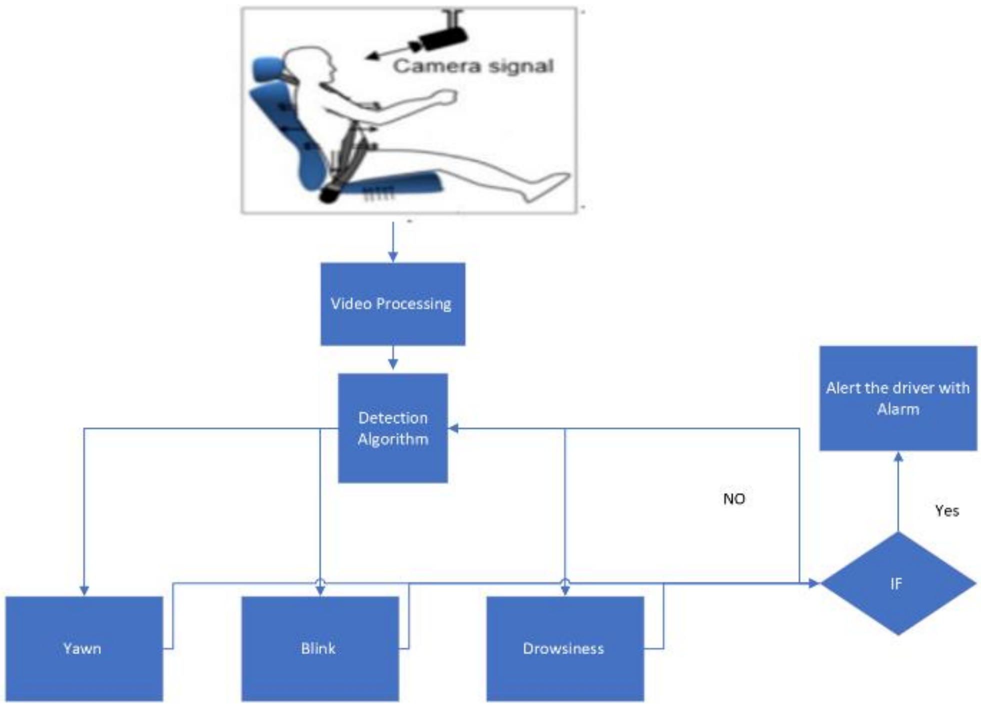
Various views (logical, process, physical, development) of architecture are presented with descriptions.

Views are one of the core components of a project which describes the architectural flows of the project which involves various stake holders in the project. It also helps to understand the intensive flow of the project. Different models have different meanings to what they stand. Based on the different UML types present in the project or discussed in the project it shows the various stake holders' involvement in the project and how they view it and control the system.

1. **Logical View:**
   The Logical View describes the various understanding of functionality that are present in the project



2. Physical view:



3. Process:
   To understand the basic flow of the project process view. What are the functionalities of the project and what they are doing and how they connected to each other.

3.2 Description for Components

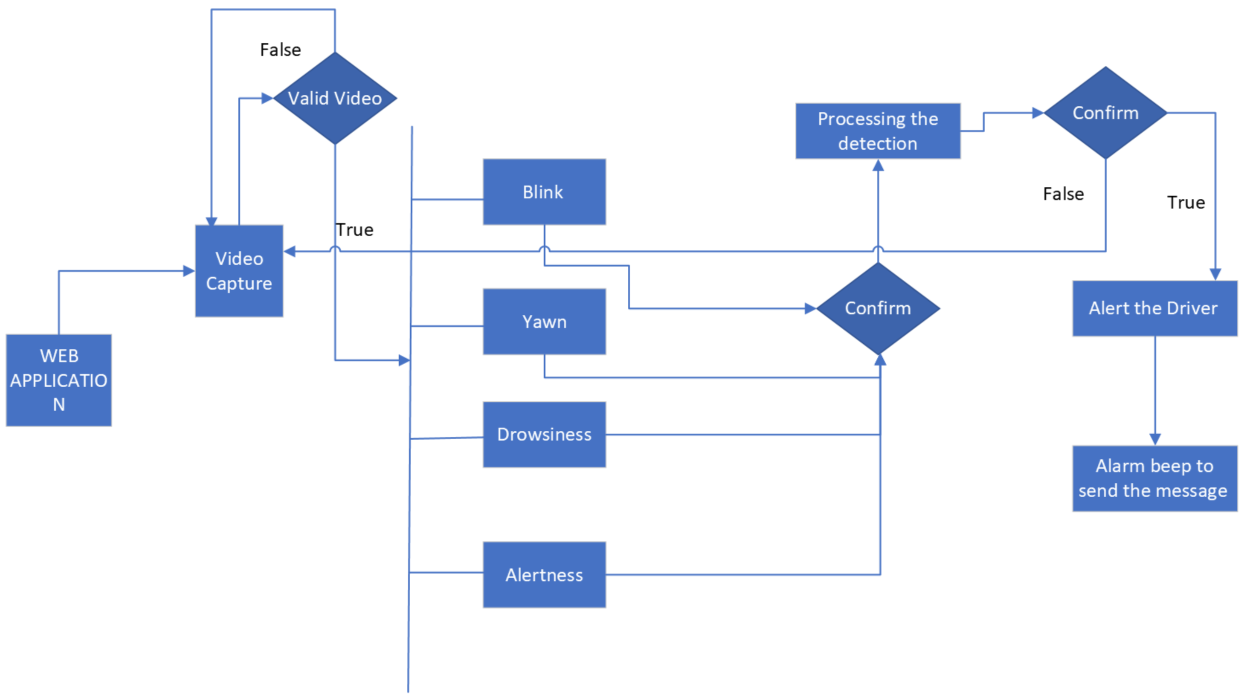
Interface description:

The input is from the video that is being captured from the web application connected to the web camera of the laptop. It detects the drowsiness of the person and the blinking rate of the person who is in front of the camera.

The output of the project is to alert alarm, which is detected, and the user is alerted to be cautious and be attentive about the situation he/she is in.

Dynamic models

State Diagram



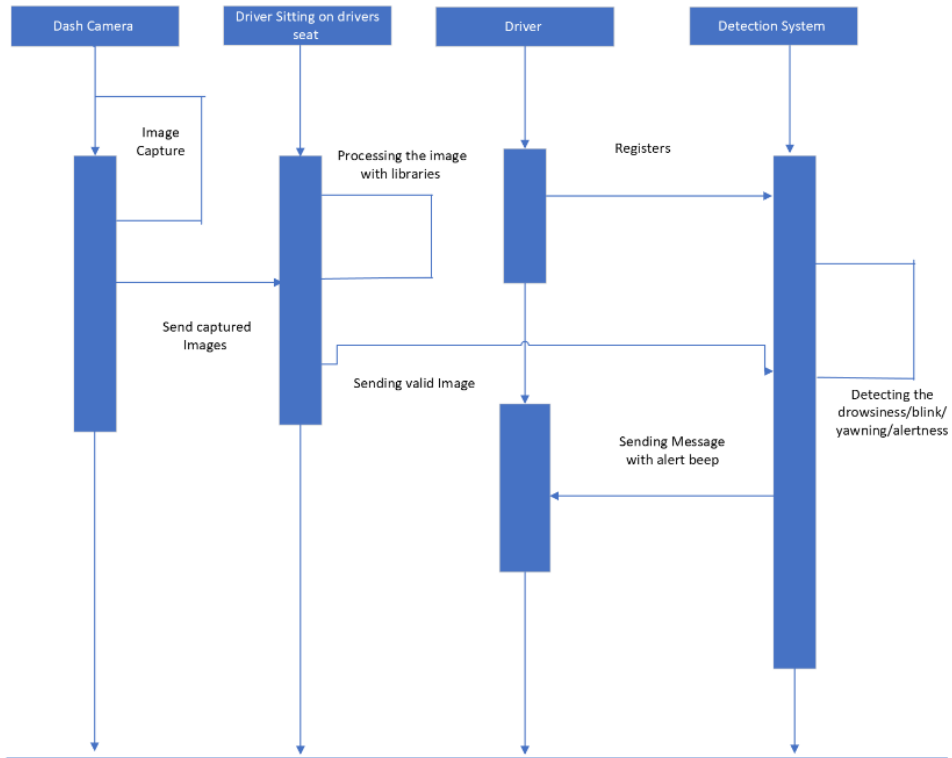
Video Capture: The video is captured through the dash camera which is sent for the validation if the image is valid it goes through next step.

Video: the video capture is the blink rate, yawn of the driver, drowsiness of the driver and alertness of the driver.

Processing stage: the image is processed to check if the image is valid for anyone of the test cases such as blink, yawn, drowsiness, or alertness. If none of the conditions are matched it will again go to the capture stage.

Alert the driver: when the image is processed and if the image is valid alert the driver. The driver is alerted with the help of an alarm.

Sequential Diagram :



3.3 External Interface Description

The software's interface(s) to the outside world (other software or hardware systems) are described.

The application is a web-based application which is connected to the web page that we have created using HTML5, CSS, JavaScript and flask and various other web technologies. Which makes it dynamic and interactive with the users and is dynamic in terms of looks and experience. The outside world will only have the camera and other functions of how it works and the functional part of the applications.

4.0 User interface design

A description of the user interface design of the software is presented.

The user interface is divided into 2 parts one part of the interface speaks about the information and reads out the details about the key features of the project such as the alarm system, Silent Monitoring, minimalistic design, and the team that was involved in making the project.

The second part of the project is the technical functionality of the project where the user video is captured, and he/she is alerted based on the drowsiness and blinks based on the video that is being captured by the web cam.

The third part of the interface is both the technical front and back front of the project where the user or the administrator can decide the parameters for the blink rate and drowsiness.

4.1Description of the user interface

The user interface for this program is the screen on the PC where user can view the output and program processing. There will also be a webpage with basic HTML, CSS and JavaScript that uses Flask to render the python code into the website.

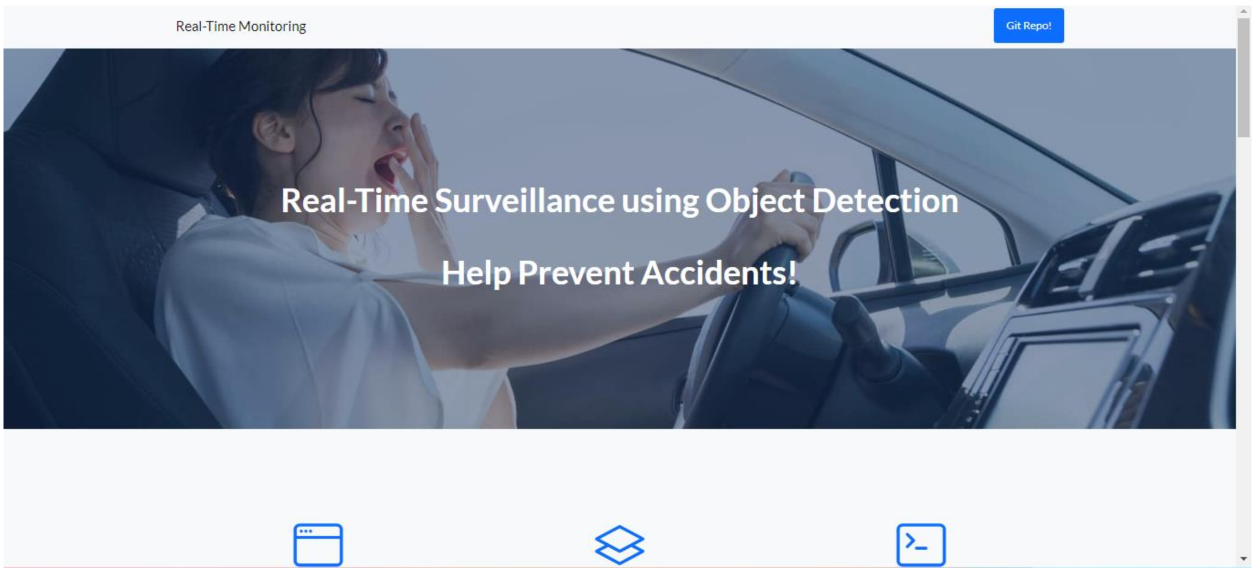


Image 1: This is the landing page of the web application which gives out the information about the topic of the project.

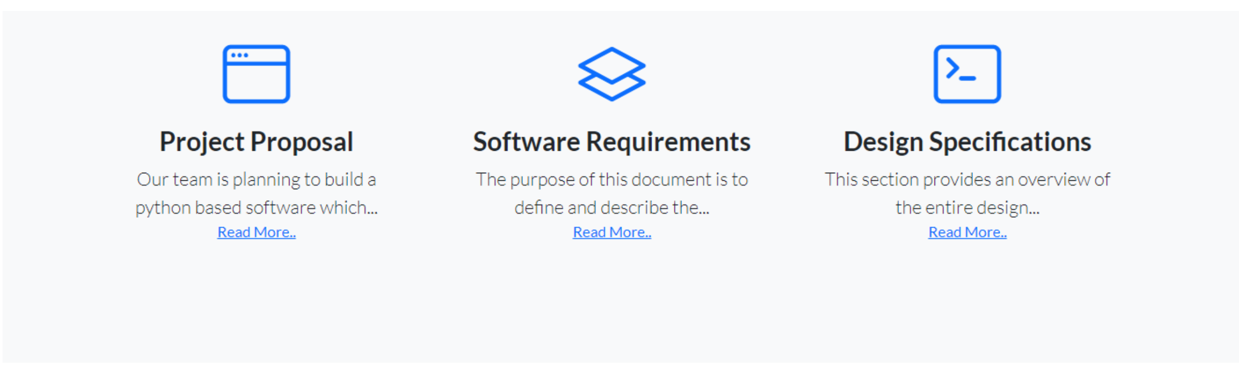
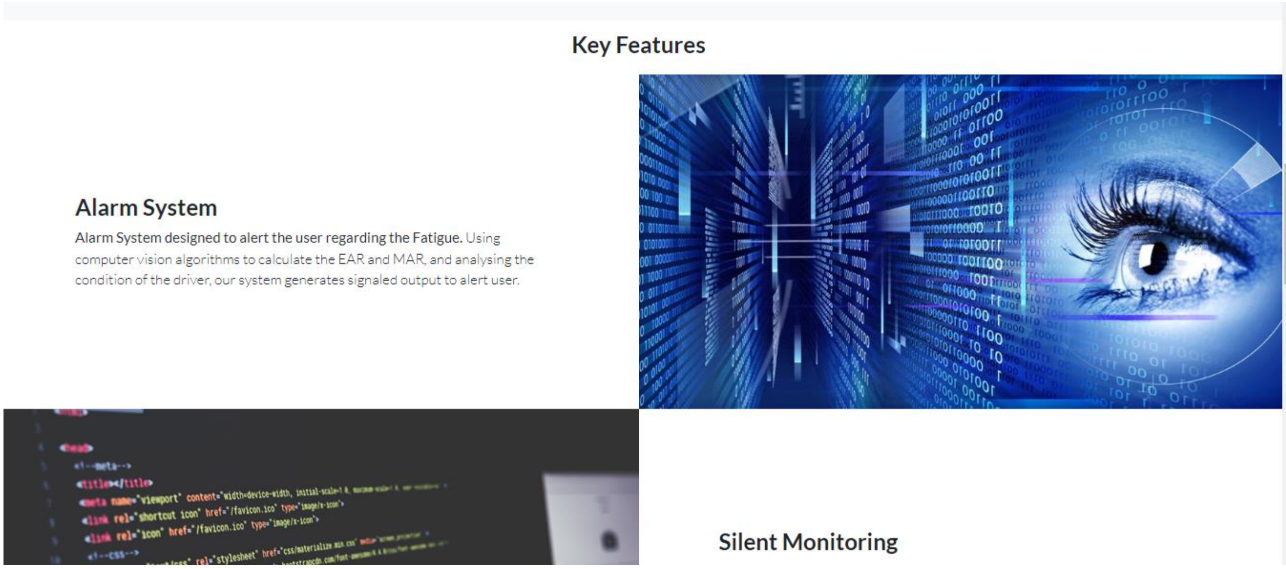


Image 2: This image has links to our various documentations on different stages of the software development life cycle.



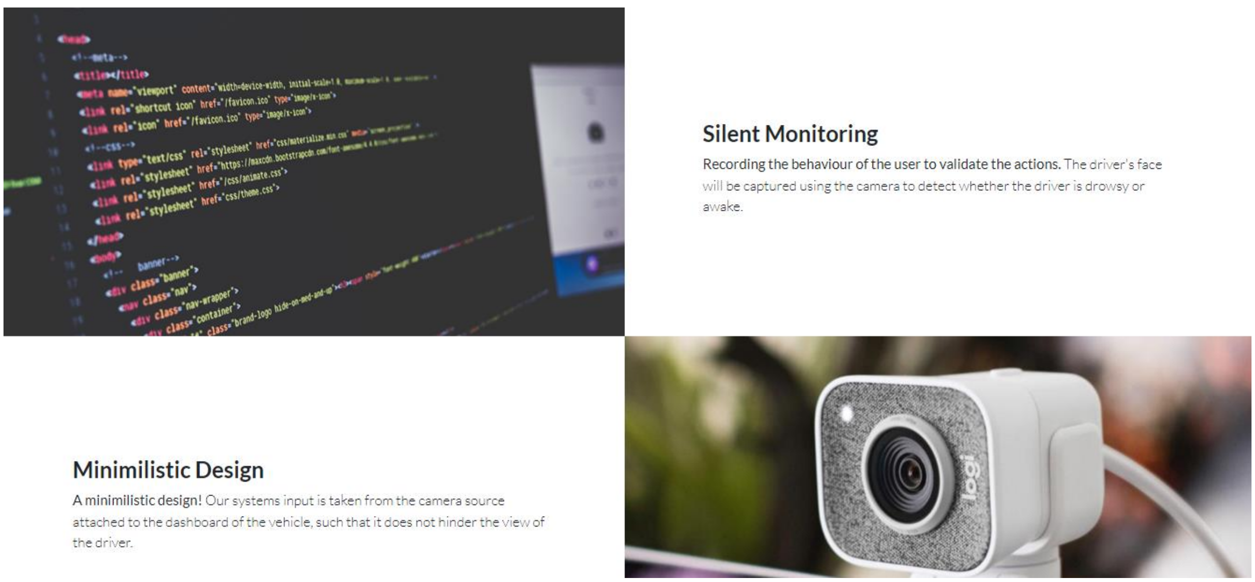


Image 3 & 4: The above two images speak about the key features and functionality of the project such as the silent monitoring and how minimalistic our system design is.

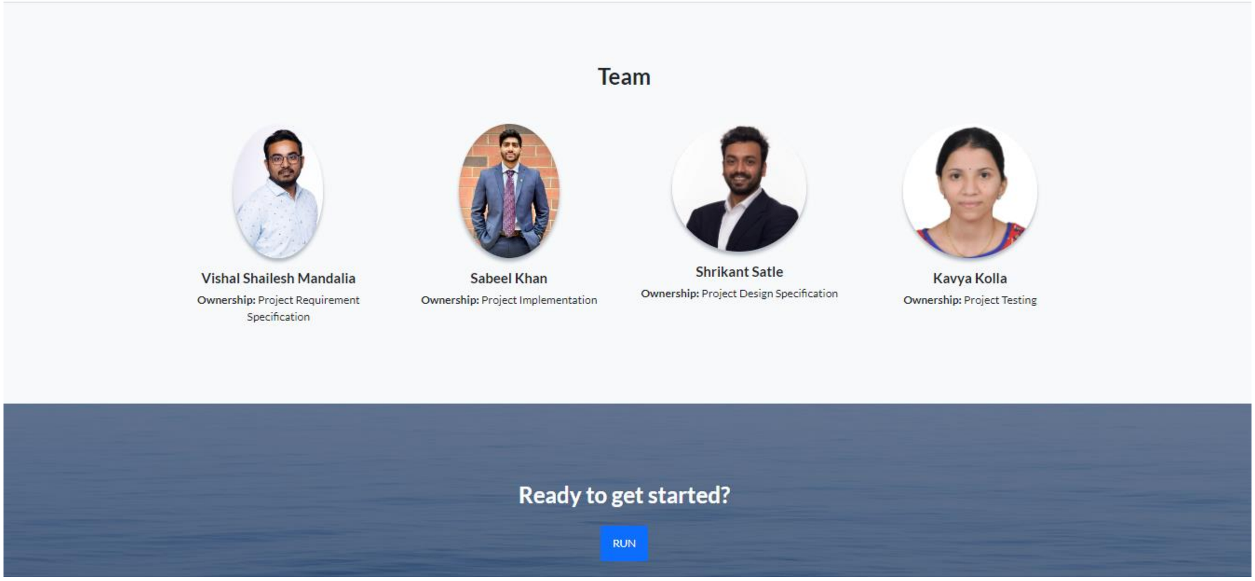


Image 5: The above image has two parts, one being our team description and what roles we play in this project as well as the “Ready to get started?” integrated button which will lead to the GUI of our software.

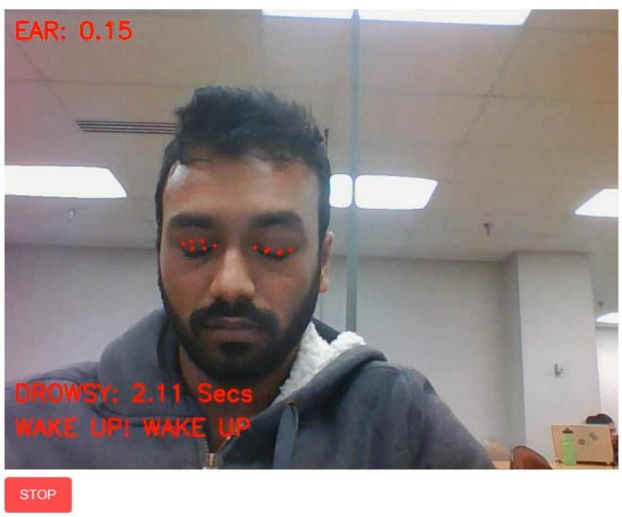


Image 6: This Image represents drowsiness and gives the alarm to alert the user

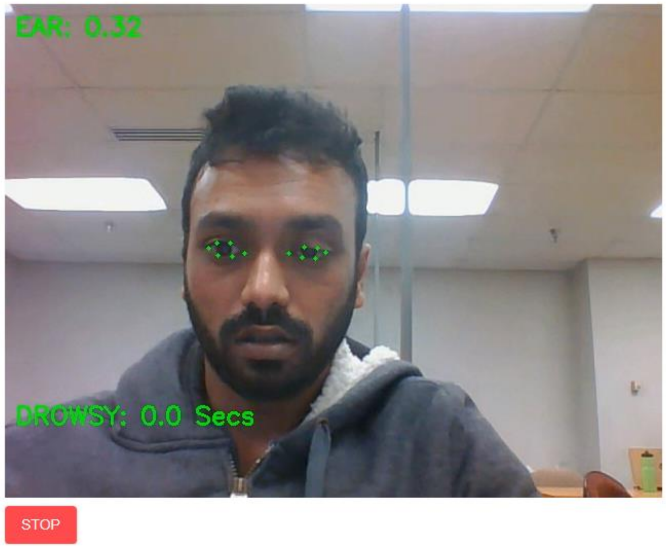


Image 7: This image gives us the details of the real-time video capture and shows the landmarking as the user is awake.

4.2 Interface design rules

Conventions and standards used for designing/implementing the user interface are stated.

1. Algorithmically simple and easy to implement. We chose to rely on off-the-shelf solutions for most stages, based on the popularity and success of the associated algorithms (e.g., Viola-Jones face detector, Support Vector Machine classifier).
2. Easily portable to different platforms. The application must run on a web application (e.g., Any static website) mounted on the vehicle's dashboard or in the current case our laptop's webcam.
3. Computationally non-intensive. Since (near) real-time performance is required, algorithms must be optimized to ensure continuous monitoring of driver's state without excessive burdening of the device's main processor.
4. Accuracy. One of the main challenges of designing such a system is related to the fact that both type I and type II errors are highly undesirable, for different reasons: type I errors (false positives) will annoy the driver and reduce their willingness to use the system (due to excessive false alarms), whereas type II errors (false negatives) can have literally catastrophic consequences and defeat the purpose of the entire system.

5.0 Restrictions, limitations, and constraints

Special design issues which impact the design or implementation of the software are noted here.

1. Lighting conditions. Frequent and drastic changes in darkness or brightness of a scene (or part of it), which may happen even during the shortest driving intervals, have been proven to be a significant challenge for many computer vision algorithms.
2. Camera motion. Poor road conditions as well as a more aggressive style of driving can introduce a significant number of vibrations and discomfort to the driving experience. Those vibrations can be passed onto the camera and cause distortion in the images which can significantly skew the results and decrease the overall performance of the system.
3. Relative positioning of device. The camera must be positioned within a certain range from the driver and within a certain viewing angle. Every computer vision algorithm has a "comfort zone" in which it performs the best and most reliably. If that comfort zone is left, performance can be dropped significantly.
5. Driver cooperation. Last, but certainly not least, all driver drowsiness detection systems assume a cooperative driver, who is willing to assist in the setup steps, always keep the monitoring system on, and take proper action when warned by the system of potential risks due to detected drowsiness.

6.0 Appendices

Presents information that supplements the design specification.

6.1 Requirements traceability matrix

Updating this as our project progresses gradually, will add this to our version2 of design document.

6.2 Implementation issues

1. Software environment: Currently we are developing our software in a virtual environment as many advanced python libraries cannot be installed without this, but this may cause some implementation issues to be operable on different platforms.

2. Eyewear/Glare: When testing we came across a major issue which can render our project accuracy as in wearing spectacles and if a reflection or glare hits the spectacles, our software lacks accuracy to detect drowsiness in the user.