Bilkent University

Department of Computer Engineering

Senior Design Project

SAVE: safe drive

High Level Design Report

Elif Demir, Utku Kalkanlı, Büşra Ünver, Celal Bayraktar, Münevver Uslukılıç

Supervisor: Dr. Hamdi Dibeklioğlu

Jury Members: Dr. Can Alkan and Dr. Çiğdem Gündüz Demir

December 27, 2020

1. Introduction

Everyday people use transportation to reach a destination. After the Covid-19 pandemic, people try to avoid using public transportation. As a result, private vehicle numbers in traffic increased significantly. More individual vehicles mean more errors in traffic. Most road accidents occur due to human error, and one of the most common types of error is driver fatigue. Traffic Accident Commission of Australia states that 20% of fatal road accidents involve driver fatigue. Furthermore, 30% of the severe crashes again involve driver fatigue [1]. Additionally, the driver's medical condition is also important in traffic. A study done by the Centre for Automotive Safety Research, University of Adelaide shows that 13% of the crashes investigated are caused by a medical condition [2].

Car companies supply some features in their cars to decrease such accidents caused by human error. However, the cost of those cars is not affordable in general and cars produced previously do not benefit from the latest developments in technology. Considering all these problems, we tried to come up with a solution to decrease human error factors in the traffic and to make the latest driver assistant features available for everyone. As a result, SAVE came up.

In the development process of the idea, we conduct market research on similar products on both vehicle and mobile application markets. After research, we find out there are similar products in the mobile application market like DRIvision [3] and car manufacturers already implement similar driver assistance applications on the vehicles. However, SAVE differs from these applications. Unlike DRIvision, SAVE mainly focuses on the driver along with the road related warnings. Although there are vehicles that provide similar features, we aim to supply these features at a much lower cost to drivers.

As mentioned before, the main focus of the SAVE will be on driver behavior. Additionally, we also provide other driver assistance like the crash emergency message and more. The features of the application will be explained in detail in the following sections.

1.1 Purpose of the system

SAVE is a mobile driver assistant application. The main purpose of the system is supplying additional security measures to the vehicle which mainly focuses on the driver behaviour. SAVE aims to prevent accidents by tracking the eye motions, mood, health condition and driving pattern of the driver using the various computer vision and machine learning models. Additionally in case of any accident detected by the application, an emergency message will be sent to people on the emergency list if the driver does not respond to the application.

1.2 Design goals

1.2.1 Usability

- This Mobile application should be easy to use and easy to understand how to use its functions to be user friendly. This application should have small and accessible buttons to navigate.
- As the users are drivers, they can interact with this application using buttons that are accessible within 3 seconds or with voice control so that they don't have to view the application screen while driving.
- To not distract the driver application should have warm colors and a night mode that users can choose to use.
- The application should use both front and rear cameras.
- The application should be compatible with all android phones above Android 6.0.0.
- For best performance application should be used with a phone holder stick to the car window.

1.2.2 Reliability

- The application's main functions such as drowsiness detection should be used without internet requirement.
- The application can be used during night driving on a road with street lights.
- The application can be used with glasses, the reflection on the glass should be noticed and ignored by the application.

1.2.3 Performance

- The application should react to the user within seconds to inform them of any behaviors that are tracked by the application, it should not be greater than 3 seconds.
- The application should assist the driver all the time during driving.

1.2.4 Security

- The application should protect the user's personal data from any cyber-attacks using encryption.

1.3 Definitions, acronyms, and abbreviations

Android: Google's mobile operating system.

Drowsiness: Difficulty keeping eyes open and feeling heavily tired while driving.

Driver Assistance: Helping the driver of the vehicle by offering some useful features.

Remote Photoplethysmography: Remote PPG

mouth aspect ratio: MAR

eye aspect ratio: EAR

Python framework for Virtual Heart Rate: pyVHR

Amazon Cloud Server: AWS

Machine Learning: ML

1.4 Overview

SAVE will be an Android application for smartphones that will assist drivers. The application will detect drowsiness, heart rate, and emotion of the driver and will send an emergency message in case of a collision. The drowsiness detection system will warn the driver if the eyes are closed longer than a blink. Also, the system will track the number of eye blinks and yawns, and heart rate. If there is an abnormal situation, again the system will warn the driver of drowsiness. In addition, if the telephone is connected to Bluetooth and the driver confirms the system will open upbeat music to keep the driver awake. Heart rate will be tracked while driving and in a situation where the drivers' health condition rates are extremely abnormal, SAVE will show appropriate health centers, hospitals via Google Maps and create a route. By voice output, it will ask the driver whether they can drive to these locations, if not it may make an emergency call. Moreover, the application will track emotional state and in case of anger, the application may open a relaxing song to reduce the temper if the telephone is connected to the car via Bluetooth. Another feature of the application will be the collision emergency message. By measuring G force in the car during the trip, the system may detect if a collision occurs by an enormous change of G force. If the system detects any collision, it will ask the drivers about their situations. If the driver does not respond, the system will send an emergency message to the people on the emergency list which are determined by the users.

Additionally, if we have enough time, we are planning to add a pedestrian warning system, traffic light assistant, distance tracking assistant, and distraction detection system. The pedestrian warning system will warn the driver if there is a pedestrian on the sidewalk and the warning level will increase if the pedestrian jumps onto the road. The traffic light assistant will warn the driver to move if the traffic light turns green. Distance tracking assistant will check the distance between the driver and the car in front and if the driver is too close than the usual following distance according to the speed of the car. The distraction detection system will track the eye movements of the driver and will warn the driver if the driver is not focusing on the road. All warnings given by the application will be in the form of audio and visual.

2. Current software architecture

Although there aren't any software applications that have similar features to SAVE, there are plenty of software which includes particular functionalities of our app, all of them working on desktop. In this section, we will reveal the modules and functionalities which are relevant to our features that we have searched.

2.1 Face Tracking Using ML Kit

ML Kit's face detection API enables us to detect faces in an image, or video and identify key facial features, and get the contours of detected faces. By this, we can get the information we need to perform tasks which includes driver expressions.

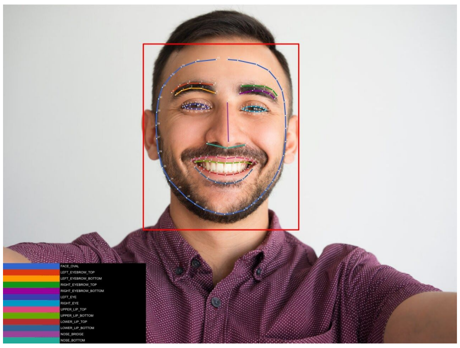

Figure 1: MLKit Image

2.2 Remote PPG Using pyVHR

pyVHR is a comprehensive framework for studying methods of pulse rate estimation relying on remote PPG. The framework uses eight well-known remote PPG methods, namely *ICA,PCA,GREEN,CHROM,POS,SSR,LGI,PBV* are evaluated through extensive experiments across five public video datasets.

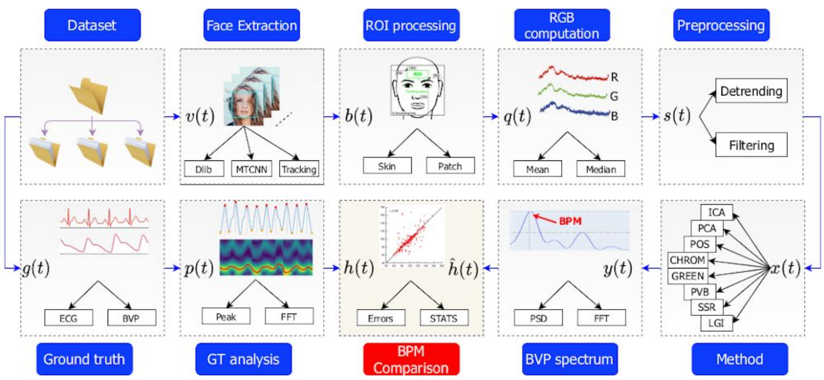

Figure 2: rPPG using pyVHR

2.3 Emotion Detection Using EmoPy

EmoPy is a python toolkit with deep neural net classes which aims to make accurate predictions of emotions given images of people's faces. It uses publicly-available datasets Convolutional models built using Keras framework.

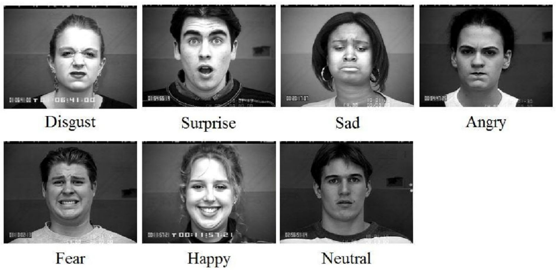

Figure 3: Emotion Detection using EmoPy

3. Proposed software architecture

3.1 Overview

In this part, we will firstly explain subsystem decomposition, where the subsystem structure of our system is described in detail with diagrams and classes. Then, we will provide hardware/software mapping of the system which shows the allocation of resources in our project. Then we will explain the persistent data management. We will also explain access control and security which defines the access boundaries of the users and security management in our system. Then we will explain global software control with the illustration of general flow in our system. Finally we will mention the initialization, termination, and failure conditions.

3.2 Subsystem decomposition

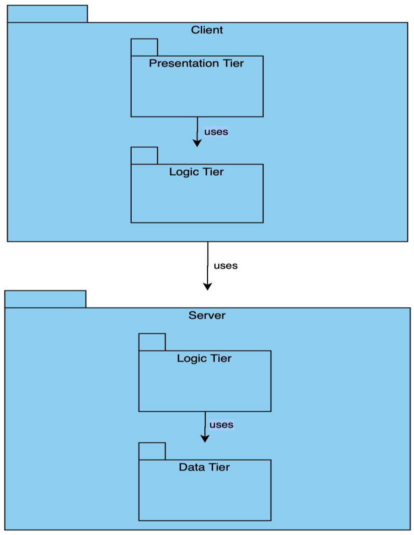

Figure 4. Three Tier-Client Server Architecture of SAVE

While giving design decisions, we wanted to have a modular system which can provide independent development process and reliability. Thus, we divided the system into packages and classes to achieve modularity. Three tier architecture of pattern is used with the intention of upgrading or replacing any tier with minimal problems in future. d with account information details or storage of machine learning models.

Presentation Tier: At the top of our application, we have a presentation tier which displays information related to computations done in a local phone or cloud based server. Main role of this tier is to give notifications to the user according to application features such as fatigue warning, or health situation warning. Also this tier handles settings and basic user operations, interaction with users via voice recognition during taking verbal inputs or giving verbal outputs.

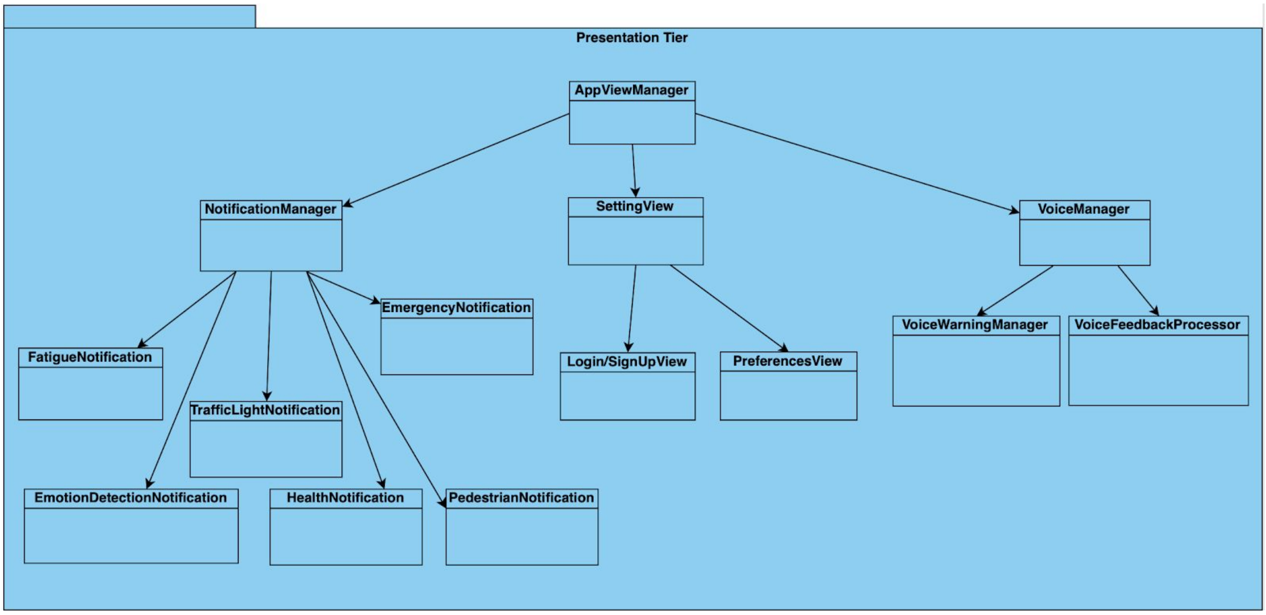

Figure 5. Presentation Tier

Logic Tier/Client: Behind the presentation layer, control of our application's functionality and performing of detailed processing is done on logic tier. Logic tier is separated in two, one of is client side and the other is on the server side. Client side of the logic tier handles basic computations on the phone such as image taking from camera, face tracking and basic image pre-processing which are common for most of our features. Also fatigue detection as an important feature of our app, runs on local devices in order to achieve its functionality even if there is no internet connection.

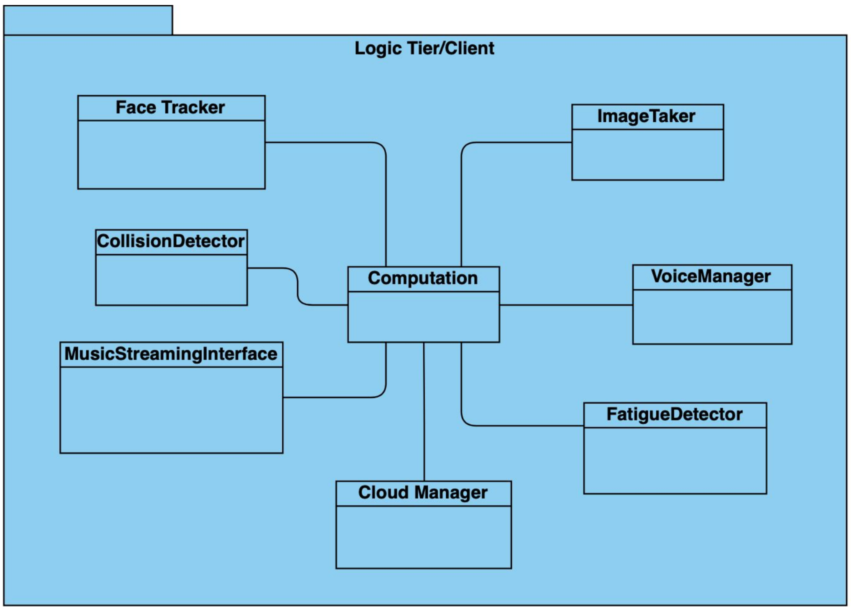

Figure 6. Logic Tier/Client

Logic Tier/Server: On the server side of the logic tier, data received from the client phone to be processed on the server is sent into Data Tier to be computed, so the logic tier of server side commands and orients the both presentation and data tiers.

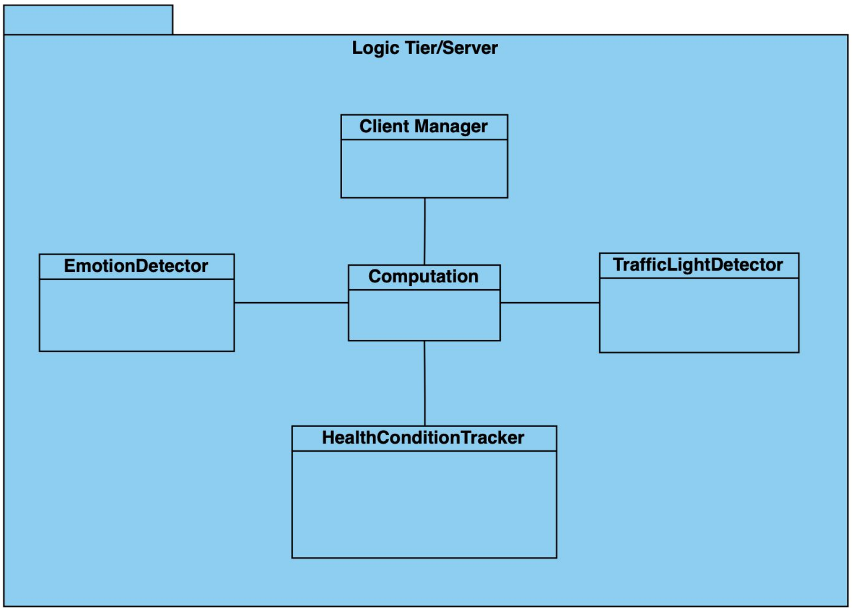

Figure 7. Logic Tier/Server

Data Tier: On the data tier we have relatively heavy computations such as remote PPG, traffic light warning system and emotions tracking. The meaningful results from server side computations will be sent to the server side and relative output will be displayed on the presentation tier. Besides that, pretrained machine learning models and account information of users are also stored in the data tier.

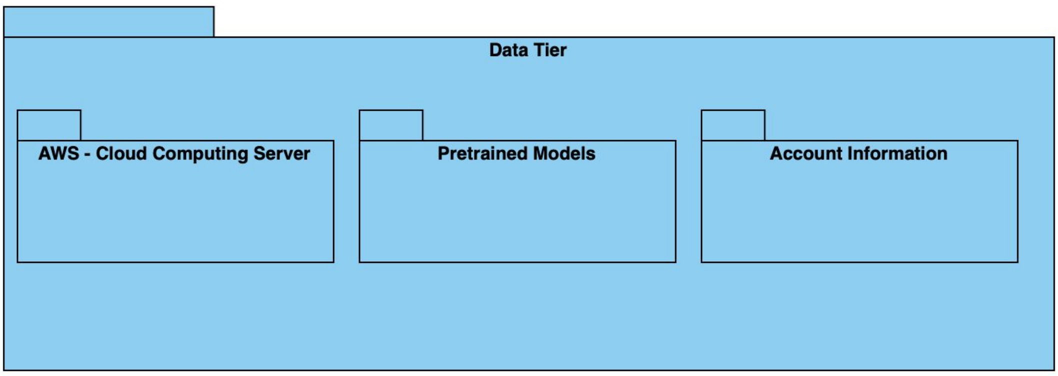

Figure 8. Data Tier

3.3 Hardware/software mapping

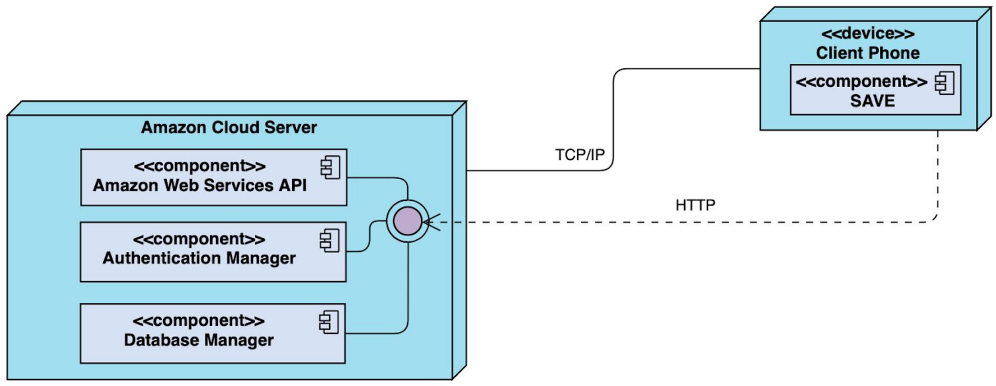

Figure 9. Deployment Diagram of SAVE

Deployment diagram illustrates the distribution of software components at run-time. It shows how hardware components exist in our system and which software components run in each node. On the client side device, our user's phone will be running SAVE software which sends and receives data from AWS via TCP/IP control protocol and HTTP on the application layer protocol. Only node outside the client phone is AWS because our application does not communicate with any other cloud server in order to achieve its functionalities. Both authentication, cloud computing and data storage are handled on AWS.

3.4 Persistent data management

SAVE does not require any database system for most of its functions. However, we use AWS EC2 Cloud's database system to store the user's id, password. User's preferences will be stored in the local storage of the smartphone used. For example, if the user wants to use fatigue detection but does not want to use emotion detection, this choice of action will be stored in local storage of the mobile phone. These will be the only data management units the application will interact with. In addition, core computer vision and machine learning models that will work on the local machine are also stored locally. Considering the lack of internet connection during the trip, some functionalities must be executed in the phone. Remaining models will be stored in the AWS EC2 server and will be executed in the same cloud server. Other than that any of the visual or audio data will not be stored after complete functioning.

3.5 Access control and security

For authentication SAVE is using AWS Mobile library's built-in sign-in UI. In order to provide security, users get a verification code via email when they register to the system. Their passwords and email information is stored in AWS in an encrypted form with other personal information required by the system such as emergency phone number.

Our system does not store the images that are taken to be used to determine the driver's fatigue and drowsiness. They are taken from the device's camera and transformed to the AWS or processed locally by the drowsiness detection system. After the process, all the visual information related to the user is deleted from all the systems.

3.6 Global software control

The control system of save is event driven. System should be able to take information from the user and use the information in it's different functionalities in a synchronized manner. The system will take both the model driven information coming from the phone's front and rear cameras, for example the distance between eyelids to calculate fatigue detection, and the user driven information such as controlling the preferences and navigation using voice or app interface. This can be implemented using trackers, as an example a face tracker that sends the information of the user to different computations, and event listeners such as onClick() or onPress() or SpeechRecongnition() for user interaction, achieving both voice and manual control. To get information from both front and back camera of the phone, system needs to switch between the cameras in a pipeline manner

Same information coming from the user can trigger multiple events at the same time, those events aren't necessarily dependent on each other's outcome. Each event has its specific subsystem for handling, and they can work simultaneously with other subsystems, taking the events that are relevant to its cause in the order they come.

3.7 Boundary conditions

3.7.1 Initialization

Users must login to our system to accept terms and conditions and consent to application recording their movements for calculation purposes. By default voice control and many tracking features except fatigue and health condition detection will be closed at initial start, users can choose what kind of assistance they want through settings.

3.7.2 Termination

Users can terminate the app by closing the application window.

3.7.3 Failure

Users can experience some errors such as internet connection failure, login failure, face recognition failure due to wrong angle or low light resource. Our aim is to minimize those errors by not depending on the internet for our main functionalities and properly inform users of what are their options regarding the technologic constraints and how they can use our application in various conditions like during the night or a place with no internet connection.

4. Subsystem services

4.1 Presentation Tier

4.1.1 AppViewManager

AppViewManager subsystem consists of the NotificationManager, SettingsView and Voice Manager. It provides the visual interaction between the application and the user via settings management and handles the notification and warning features of the system.

4.1.1.1 NotificationManager

This component consists of the notification controls of the system. The individual notification modules are explained in the following sections.

4.1.1.1.1 FatigueNotification

This module is used for giving the voice or push notification to the user when the application detects an ongoing drowsiness situation. The requirement for making this feature available is only if the user set this as allowed in the settings view. This notification will suggest near resting places to users.

4.1.1.1.2 EmotionDetectionNotification

This module is used for giving the voice or push notification to the user when the application detects a non-positive or non-neutral emotional situation. The requirement for making this feature available is only if the user set this as allowed in the settings view. This notification could suggest a music playlist depending on the mood of the user if the user activated the feature.

4.1.1.1.3 TrafficLightNotification

This module is used for giving the voice or push notification to the user when the application detects the change in the traffic light by using the rare camera.

4.1.1.1.4 HealthNotification

This module is used for giving the voice or push notification to the user when the application detects any unhealthy situation of the user while driving by using remote PPG. The requirement for making this feature available is only if the user set this as allowed in the settings view. This notification will also require feedback from user regarding their condition, if no feedback is provided, if the user activated the feature, system will inform an emergency contact of the situation

4.1.1.1.5 EmergencyNotification

Collision emergency message subsystem module to notify the dramatical change in the G force and send the GPS location to the emergency call number. This notification will also require feedback from user regarding their condition, if no feedback is provided, if the user activated the feature, system will inform an emergency contact of the situation

4.1.1.1.6 PedestrianNotification

Give warning notification to the user when there is a pedestrian standing on the sidewalk, depending on the preference of the user could be a voice warning or a push warning.

4.1.1.2 SettingsView

This module is in charge of managing personalization and various settings of the application. Users can handle the notification permissions by this component.

4.1.1.2.1 LoginSignUpView

Module to present the user a login/signup view and send the form information to the system's cloud for registration/logging in.

4.1.1.2.2 PreferencesView

Module to handle the preferences of the user, like which kind of tracking do users want from the application or which interaction/feedback system do they want, along with various personalizations like additional health condition tracking or UI preferences.

4.1.1.3 VoiceManager

Subsystem module to give audio warnings and process the user feedback in audio form, if the user chooses to interact with the application through voice control or give feedback to the system's health condition checks in audio format.

4.1.1.3.1 VoiceWarningManager

This module is used for sending a voice message to the user in situations that there is a fatigue, unhealthy or any unstable emotional condition.

4.1.1.3.2 VoiceFeedbackProcessor

This module takes the reply message of the user to understand the commands in the situations that the application sends a voice message in order to call someone, play music or send GPS location.

4.2 Logic Tier

This subsystem consists of two main logic tiers that are for client and server systems.

4.2.1 Computation

This class handles the computation by the information coming from the following modules. It handles the control flow of sending notifications, voice messages and warnings to the user.

4.2.1.1 ImageDetector

This module connects the phone camera and gets the image of the user and the read to do the computations with those captured images.

4.2.1.2 FaceDetector

This module detects the user's face and determines the specific points on the face to do the computations based on the gestures and skin tone of the user while driving.

4.2.1.3 CollisionDetector

This module is used to detect the dramatical change in the G force and also gets the GPS location in case the user requires to send information as text messages to the saved phone numbers.

4.2.1.4 VoiceManager

The subsystem to include and use functionalities of voice control and recognition system with appropriate event listeners.

4.2.1.5 MusicStreamingInterface

This module is used for streaming music via Spotify if the user applies the music streaming request and also the car is connected to the music system of the car.

4.2.1.6 CloudManager

This module is used for sending packages to the cloud server and after running the packages on the server, getting the results to the logic tier.

4.2.1.7 FatigueDetector

The subsystem that runs a face detection algorithm to collect information about faces like the distance between eyelids or yawning frequency to calculate the level of fatigue the user is experiencing.

4.2.2.8 Client Manager

The subsystem that coordinates the data transaction between client and server on the server side.

4.2.2.9 TrafficLightDetector

The subsystem that uses the back camera to detect any traffic light nearby to keep track of light changes to be able to inform the user.

4.2.2.10 EmotionDetector

The subsystem that uses a face detection algorithm to get the information about the faces to calculate mood changes the user is experiencing.

4.2.2.11 HealthConditionTracker

The subsystem uses remote PPG technology to track the health condition of the user, using the front camera of the phone.

4.3 Data Tier

This subsystem consists of Cloud Computing Servers, Pretrained Models and Account Information modules.

4.3.1 AWS - CloudComputingServers

This module is used to do the computations and storing the data on cloud services of Amazon Web Server. Other than that, it is used for executing the deep learning models in the quickest way to get better results in the application.

4.3.2 PretrainedModels

This module is used to execute the pretrained models such as fatigue and emotion detection and remote PPG models that are going to be imported to the project.

4.3.3 AccountInformation

This module is used for getting the required information for signing up.

5. Consideration of Various Factors in Engineering Design

At the consideration of various factors, the external factors of the environment that the system works in will be considered. The SAVE application that is being proposed attempts to provide a better driving experience for drivers from different ranges that are not able to afford high levels of technological service.

5.1 Data Privacy

There is no use of any personal data against GDPR however, these regulations tend to change with improving data networks globally. Because the application's main focus is the driver, driver related features may violate the data privacy issues in the future in case changes occur in the definition of private data. The features related to data privacy are fatigue detection, emotion detection, health recognition, and distraction detection. Any of these features make any video recording to further analysis or sharing.

5.2 Distraction Factors

In order for the application to function optimally, the telephone must be located in a place where both the rear and front camera will see the area of functioning. Since the location of the telephone will be in the sight of the driver, visual and audio notifications from the other application may attract the driver. As a result, the focus of the driver may slide from road to telephone. Additionally, the warnings application gives to the driver also has the possibility to distract the driver. To make this distraction minimum, the sound volume of the warnings is adjustable. The driver can change the sound level to minimize the distraction. Furthermore, to minimize the physical interaction between the driver and the telephone, user interaction related features provided both audio and touch controllable such as activate/deactivate a specific feature.

5.3 External Factors

Since the application analyzes the visual input from the driver's face, the functioning of the features may be affected by the physical factors. The fatigue detection system program detects fatigue according to EAR and MAR values. EAR value changes from person to person. Even though the threshold value of EAR for the fatigue alert defined for an average value, the eye aperture of some people may be below the threshold and the system may not function well. Although wearing glasses do not affect the functioning of the application, usage of the sunglasses blocking the view of eyes from the camera, and the application could not detect the eyes and face properly. Unfortunately, with the Covid-19 pandemic, people use face masks for protection. However, similar to wearing sunglasses, using face masks blocking the view of the mouth from the camera and application could not detect the mouth properly. As a result, related features may not work properly.

5.4 Upcoming Mobile Device Designs

Constant mobile device development affects the application's working style. Development in the CPU and GPU technologies increases the capability of the processors while decreasing the physical size. Parallel to these developments capabilities of the telephones increases every year. As a result, features that work on cloud servers may work on the telephone's own hardware. Position and the resolution of the cameras also open the change over the years. Telephone location may be required to calibrate for optimum performance in case of telephone change. Telephones are getting bigger in time. New telephones may not fit old phone holders. Users may purchase a new phone holder to locate the telephone.

5.5 Evaluation of the Constraints

	Effect Weight (out of 10)	Effect on the Application
Data Privacy	5	Does not create a problem with the current GDPR. Possible data privacy definition change may require further action in the implementation.
Distraction Factors	4	Distraction level changes driver to driver. More customizable user interfaces may be developed.
External Factors	9	External factors may heavily affect the functioning of the application.
Upcoming Mobile Device Designs	6	Additional hardware purchase may be required. (Additional cost for user)

6. Teamwork Details

6.1 Contributing and functioning effectively on the team

We have worked as a full-stack developer team since the beginning of the fall semester. In order to work efficiently on our project, we have scheduled meetings regularly. In our meetings, we have discussed our ideas, analyse and design issues, the progress of our project, and also distributed the required tasks for the upcoming parts of the project. Other than our group meetings, we have also scheduled meetings with our supervisor. In those meetings, we have determined assignments as the team and discussed the issues we have encountered during our work. For the purpose of having responsibilities balanced and taking the lead role in tasks as individuals, we have divided the project requirements into five main parts. We have distributed those parts according to our previous experiments and volunteering. In the reports, all of the team members were responsible.

6.2 Helping creating a collaborative and inclusive environment

We have worked both as individuals and subgroups during the project. All of us have one main task for the first semester and make progress with them. During the project implementation, we have worked together since different parts of the implementation requires another task so all of us were included in various parts of the project. While working in subgroups, we shared the leading role according to task requirements and capability of the team members according to our main tasks and solved the software design issues together.

6.3 Taking lead role and sharing leadership on the team

Büşra Ünver: Responsible for developing UI, presentation tier functionalities, and binding the different logic tier functionalities in one application.

Celal Bayraktar: Responsible for fatigue detection system.

Elif Demir: Responsible from the general back-end structure of the application. Develops the android application with authentication and other functionalities and implements its connection to the AWS EC2 cloud system.

Münevver Uslukılıç: Responsible for the EC2, S3 and SageMaker Studio instances of the Amazon Web Services and run the packages on the virtual server.

Utku Kalkanlı: Responsible for searching of emotion detection, remote PPG and face tracking

7. Glossary

Remote Photoplethysmography	Remote Photoplethysmography (PPG) is a contactless way to measure human cardiovascular activity by measuring the reflection variations of the skin registered by a video camera [4]. It only requires video recording with a high-resolution camera[5].
Eye aspect ratio	Eye aspect ratio process is the combination of eye localization, analyzing the whites of eyes and determining the period of time that white region of the eye disappears to indicate the blink rate of the human by using facial landmarks[6].
Mouth aspect ratio	Mouth aspect ratio The horizontal and vertical distance of the mouth by using 2D facial landmark locations [7].

8. References

[1] *Fatigue statistics*, Transport Accident Commision. Accessed on: Nov. 21 2020. [Online]. Available: https://www.tac.vic.gov.au/road-safety/statistics/summaries/fatigue-statistics

[2] T. Lindsay, "Medical conditions as a contributing factor in crash causation." ResearchGate, May, 2018.

[3] *DRIvision*. Accessed on: Nov. 21 2020. [Online]. Available: https://drivision.wordpress.com/

[4] Verkruysse W., Svaasand L. O., Nelson J. S., "Remote plethysmographic imaging using ambient light," Opt. Express 16(26), 21434–21445 (2008).10.1364/OE.16.021434

[5] *What is RPPG?*, Noldus. 2020. Accessed on: Nov. 21 2020. [Online]. Available: https://www.noldus.com/blog/what-is-rppg

[6] Rosebrock, A., 2020. *Eye Blink Detection With Opencv, Python, And Dlib - Pyimagesearch*. PyImageSearch. Accessed on: Nov. 21 2020. [Online]. Available: https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/

[7] Kir Savaş, Burcu & Becerkli, Yaşar. (2018). Real Time Driver Fatigue Detection Based on SVM Algorithm. 1-4. 10.1109/CEIT.2018.8751886.