Drowsiness Detection Using Deep Learning

NTI AI&IOT TRACK FINAL PROJECT

Team Members

Group 2

Ahmed Al Deeb

Phone number:
01154118192

Mail:
ahmedaldiep@gmail.com

Noran Nabil

Phone number:
01097512179

Mail:
norannabil0328@gmail.com

Shahd Ahmed

Phone number:
0 111 937 4825

Mail:
shahdahmed2021@gmail.com

Salma Ragab

Phone number:
01024442788

Mail:
salma56449@gmail.com

Introduction

Part 01

Background

Overview of the project

With increase in the number of vehicles, the occurrence of traffic and accidents is also increasing gradually. Traffic collisions are a major source of deaths every year. The National Crime Record Bureau (NCRB) reports 496,762 road-related traffic collisions in all states

A computer vision system that can automatically detect driver drowsiness in a real-time video stream and then play an alarm if the driver appears to be drowsy for the driver and his manager.

The Problem

What we want to solve

The main goal of this work is to determine whether a driver is drowsy or awake.

Hypothesis

Utilization of deep learning architectures will enhance the accuracy of both face detection and drowsiness detection.

Our Data

Part 02

Data Collection

Which Datasets we used?

Driver Drowsiness Dataset:

Extracted and cropped faces of drivers from the videos of the Real-Life Drowsiness Dataset. The frames were extracted from videos as images using VLC software. After that, the Viola-Jones algorithm has been used to extract the region of interest from captured images.

yawn_eye_dataset:

A dataset of 2900 sample image divided into 4 categories closed_eye, open_eye, yawn, and non_yawn it's used to predict drivers drowsiness throw these features.

Drowsiness Prediction Dataset:

A dataset of 2900 sample image has two labels: Fatigue which tends to drown and Active which is not.

NITYMED

130 videos, captured in Patras, Greece, displaying drivers in real cars, moving under nighttime conditions where drowsiness detection is more important.

Data Sequence

The avilable three data sets that have been mentioned didn’t give us the required accuracy in this application so we tend to generate our custom data set.

Data Sequence

yawn_eye_dataset

Only the two folders of yawn and nonyawn were used.

NITYMED dataset

frames extracted from the dataset’s videos were preprocessed and used as a supplementary data for better learning

MTCNN Face Extraction

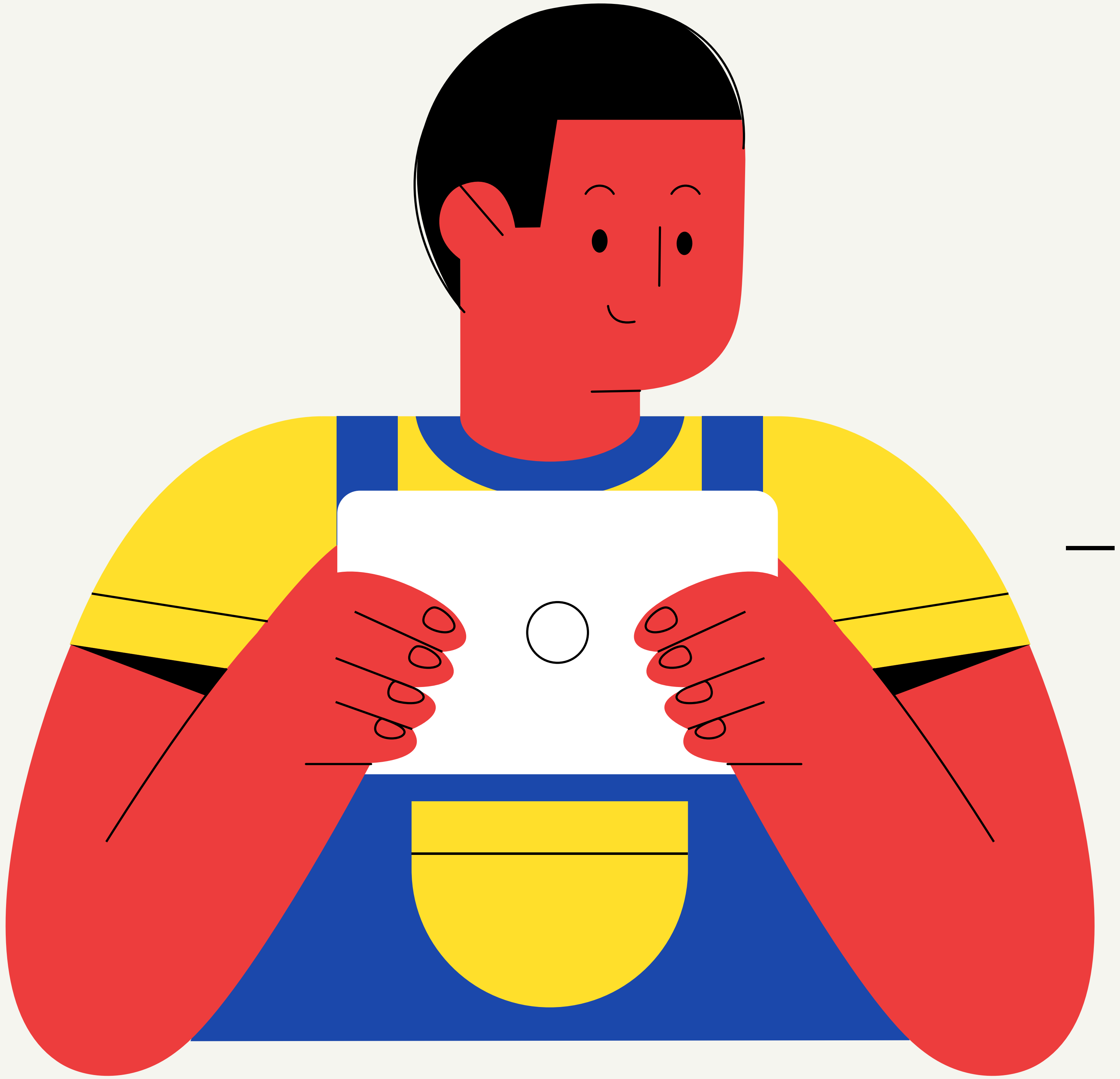
MTCNN (Multi -Task Cascaded Convolutional Neural Networks) for cropping faces from images to be easy for the model to learn.

Data Preprocessing

- Turning Video Inputs into images frames
- Face Recognition and extraction
- Resizing the images
- Data Augmentation

Data Augmentation

- Based on our experience, it is not the good choice for this data set.
- Our data is augmented and used for the model training but it results with a bad test accuracy in the real time.
- So, in our work we relyed on the original data set without augmenation.

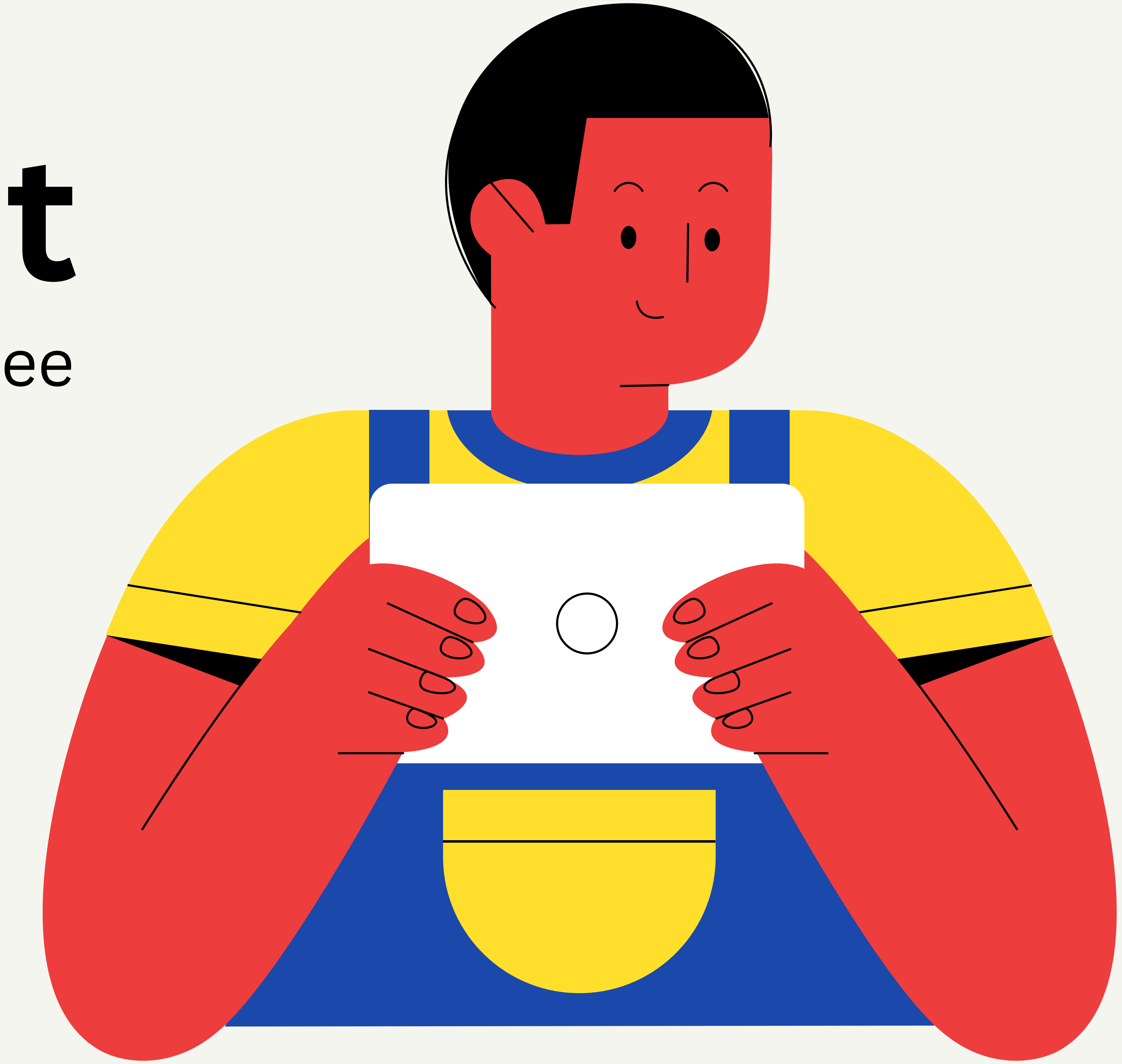


Final Data Set

The generated data set contains three classes :

- Active : 1580 Images
- Sleep :1230 Images
- Yawn : 1013 Images

Data Link



Transfer learining using different models for best model selection

Resnet50

20 EPOCHS

TRAIN:.75
VALID:0.74
TEST:0.73

Resnet152.

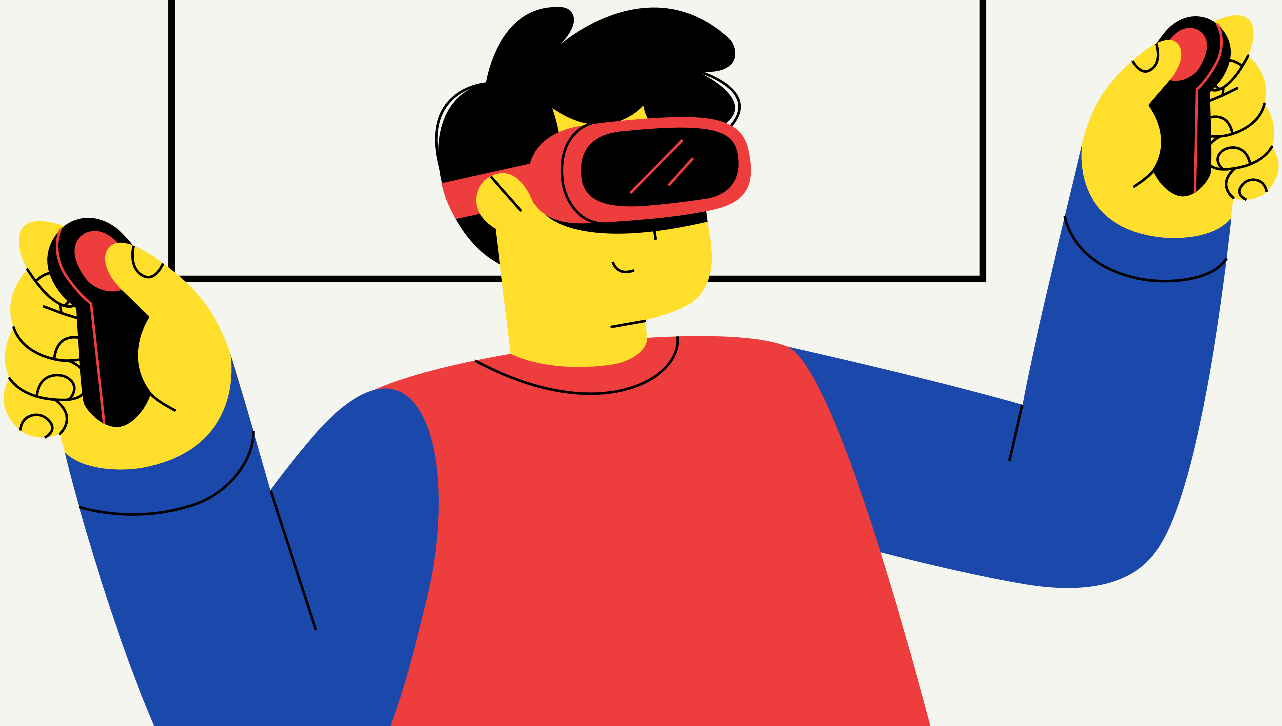
20 EPOCHES

TRAIN:0.942
VALID:0.934
TEST:0.541

INCEPTIIoNV3

20 EPOCHES

TRAIN:0.8239
VALID:0.8062
TEST:0.727



Transfer learining using different models for best model selection

DenseNet121

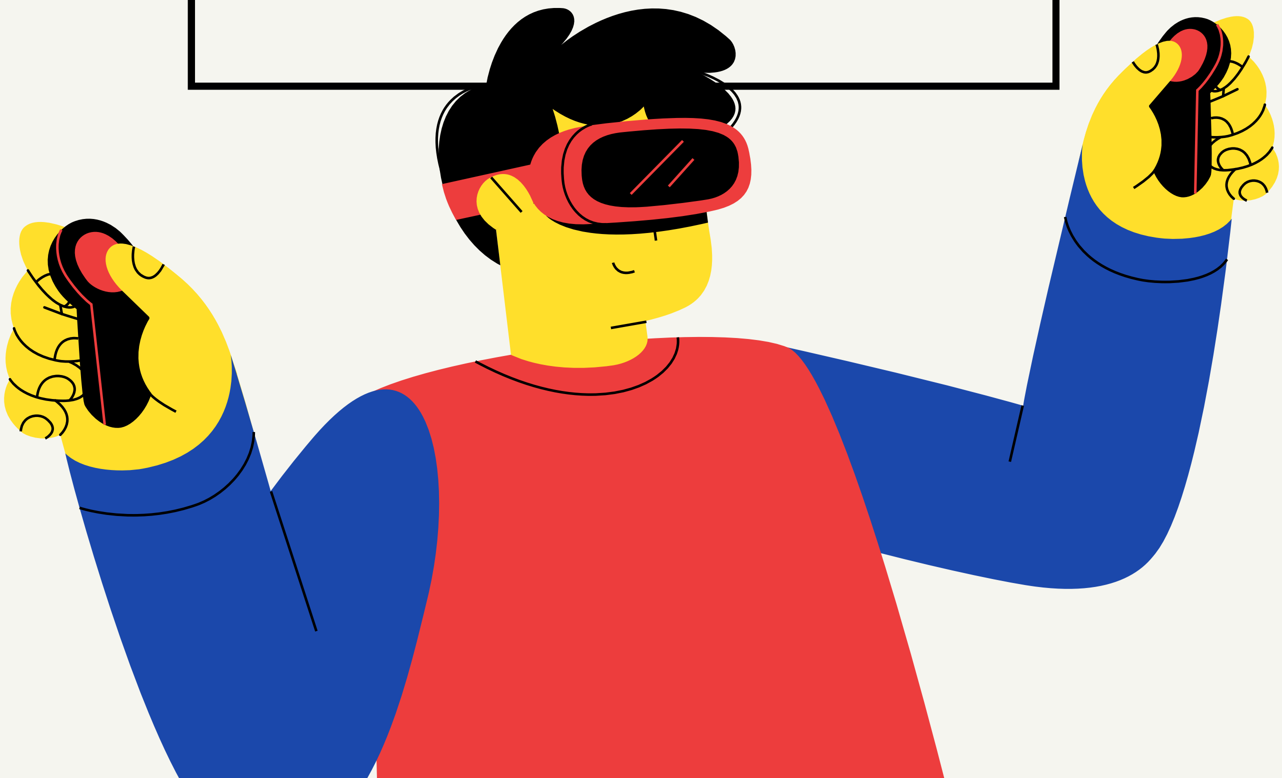
TRAIN:0.82
VALID:0.81
TEST:0.73

Xception

TRAIN:0.88
VALID:0.85
TEST:0.79

VGG16

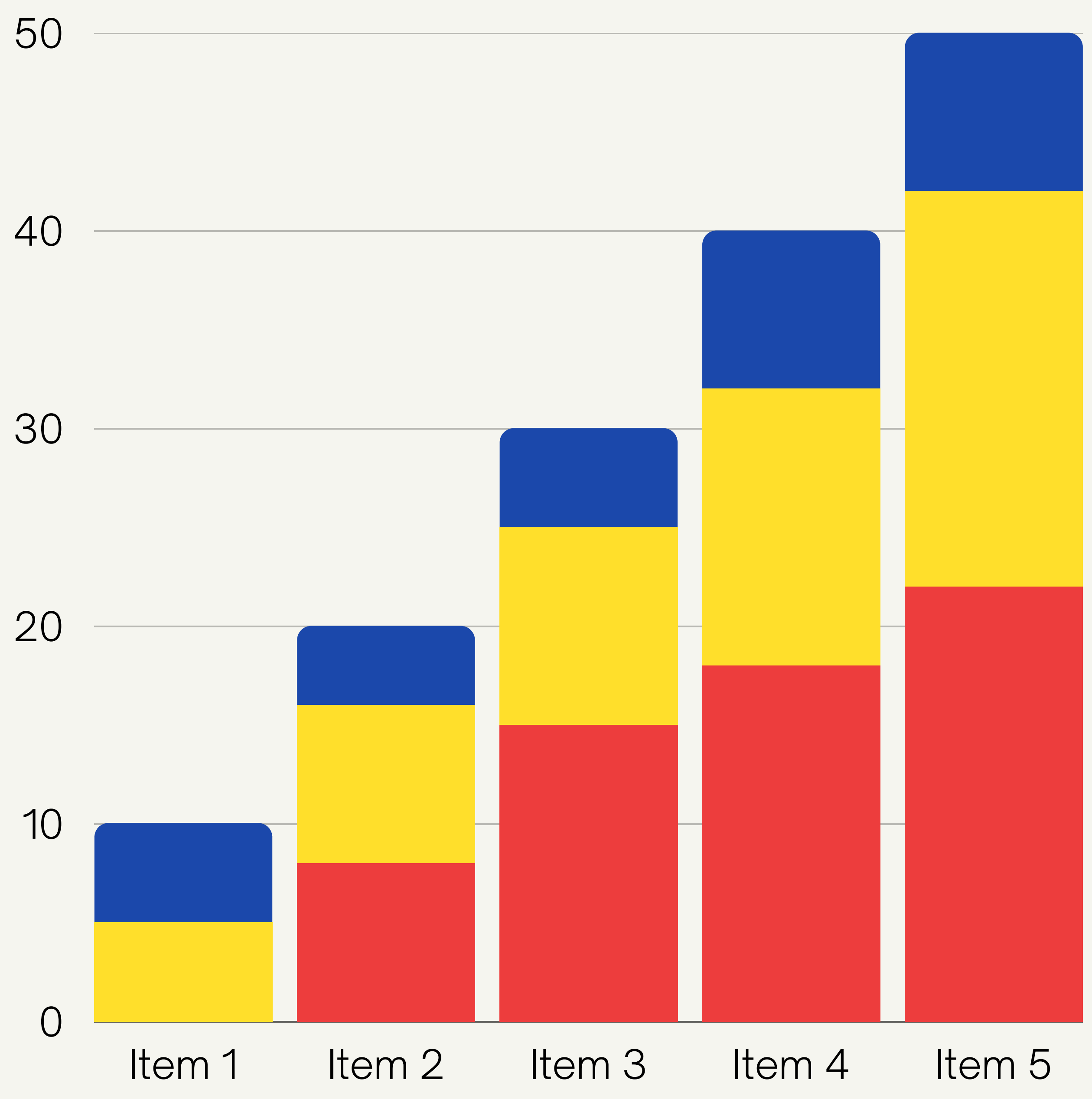
TRAIN:0.9483
VALID:0.9207
TEST:0.899



Best Model

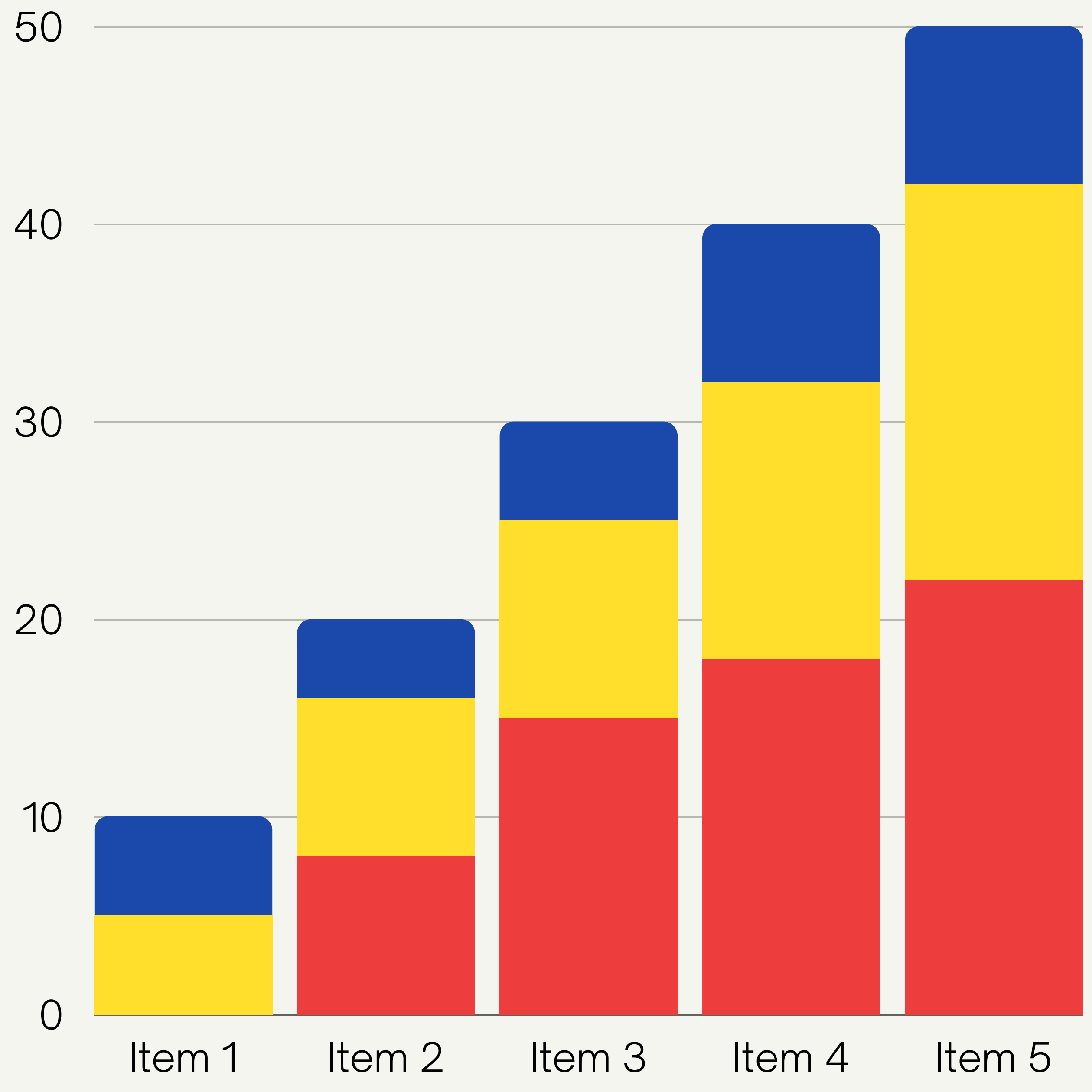
- After we use all this models we find that **VGG16** is the most suitable model for our problem with the highest accuracy.

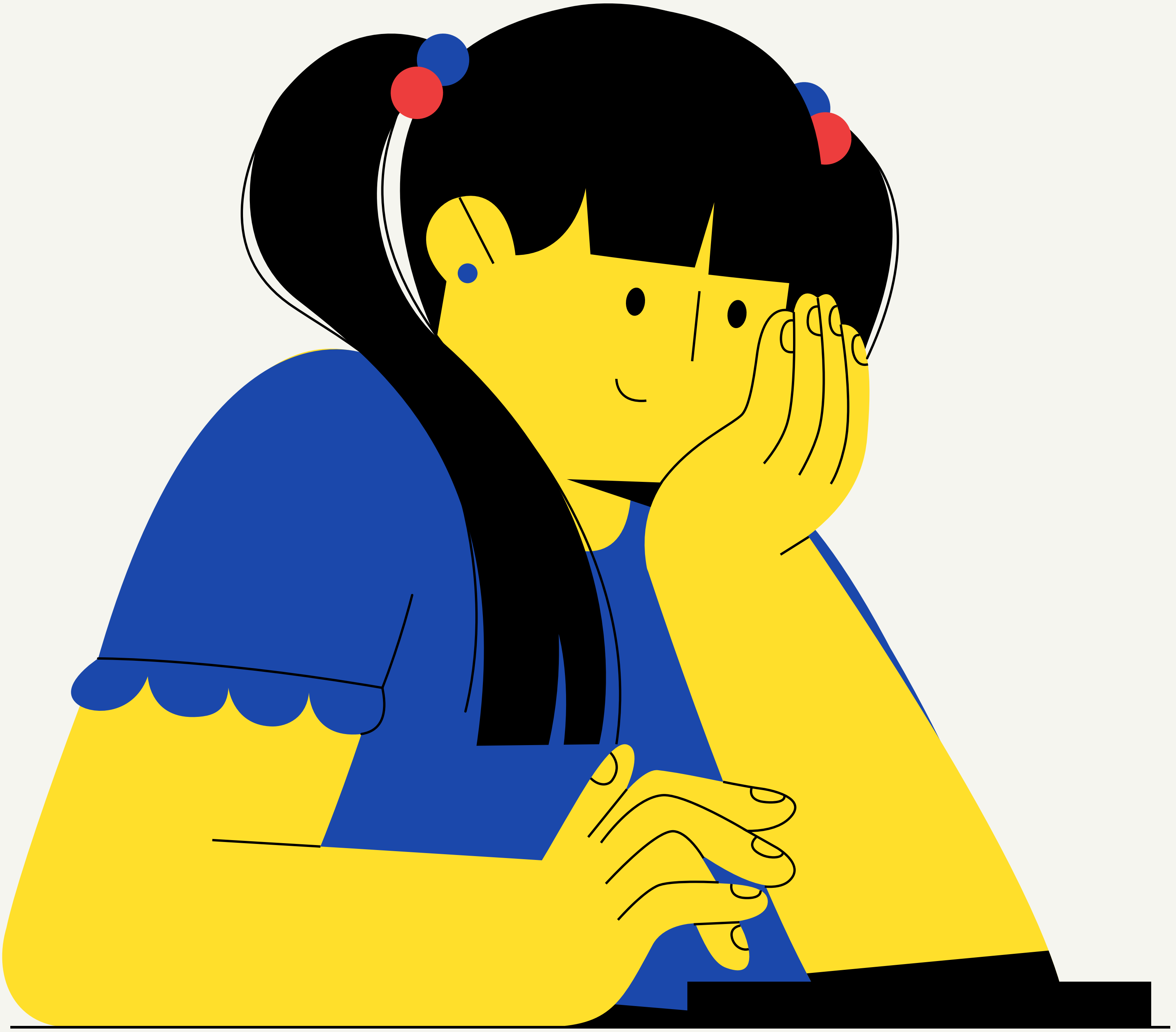
- Based on transfer learning, the model is followed by two dense layers, one drop out layer and out output layer of three softmax activated nodes.



Accuracy & Training Environment

- Training has been done on the generated data set for 30 Epochs using Google Colab free T4 GPU resources.
- Training has ended with the following accuracy:
  - Training: 99.51%
  - Validation: 94.95%
  - Testing: 96.22%





Deployment

Part 03

Realtime Processing & Prediction

Camera Usage

.Model was used to make prediction in real-time processing by accessing laptop camera though Google Colab notebook, every 0.5 second an image is token.

Your video

You can use your driving viedo for testing, five frames are token of your viedo each 3 seconds a frame is token.

Decision making

5 frames are taken from the camera then the model run on them to predict the result whether the subject is asleep, awake or going to sleep

Output

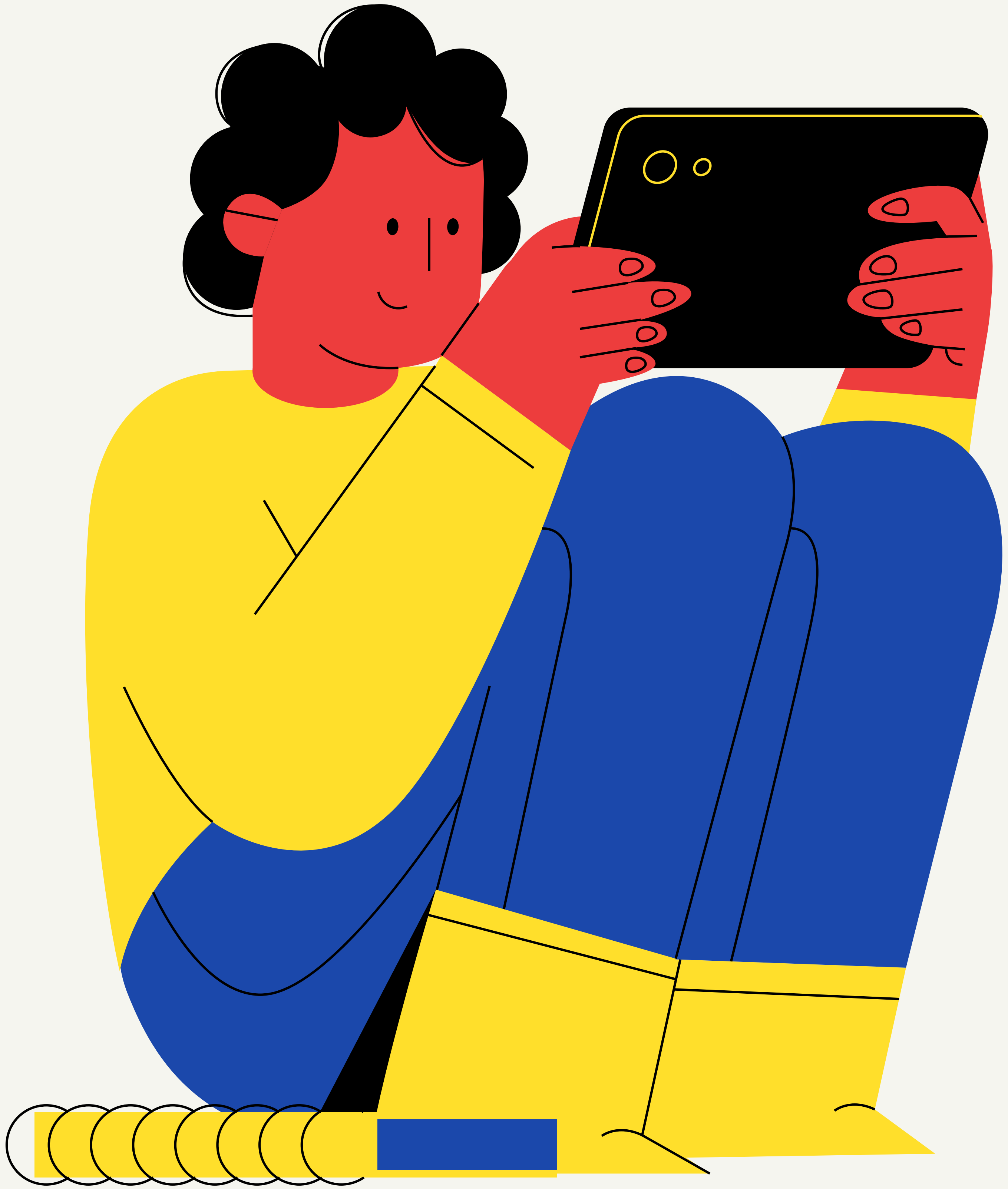
Based on the prediction a message output is generated, and sent to the driver's car to start the alarm if it is needed and to the driver manger for monitoring.

Decision Making

Drowsy driver:
sleep in two followed frames or yawning in one frame and sleep in the following frame

Driver is going to sleep :
yawning in two following frames

Awake driver :
No sleep No yawning in two following frames



IoT integration

Server Connection

The message output is uploaded on a server to a specific topic.

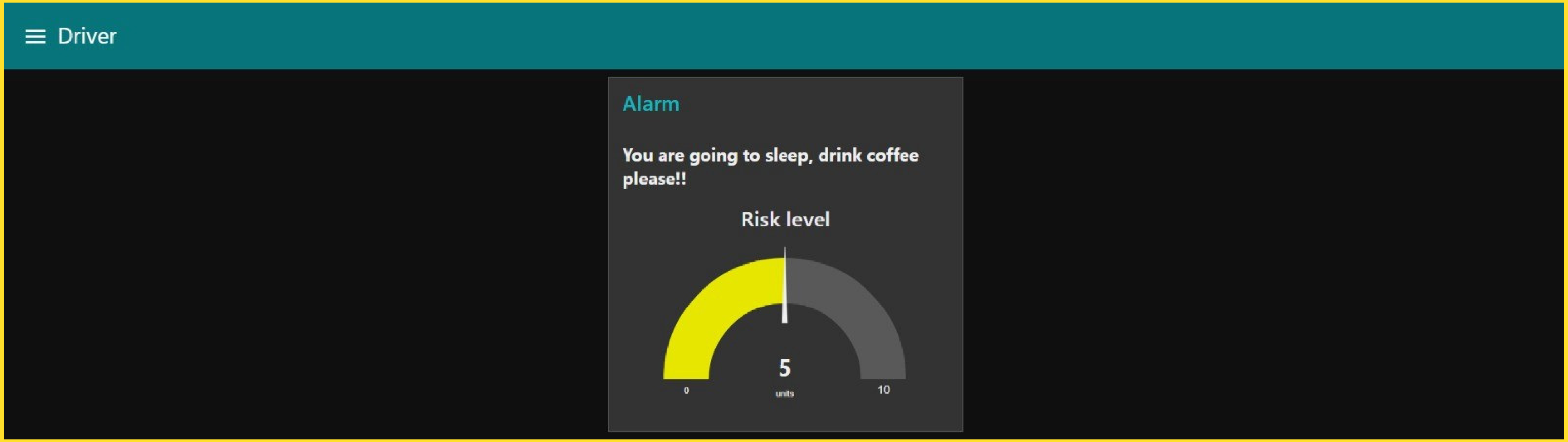
Received Message

Any IoT device can be connected to the server and receive the message from it

Action

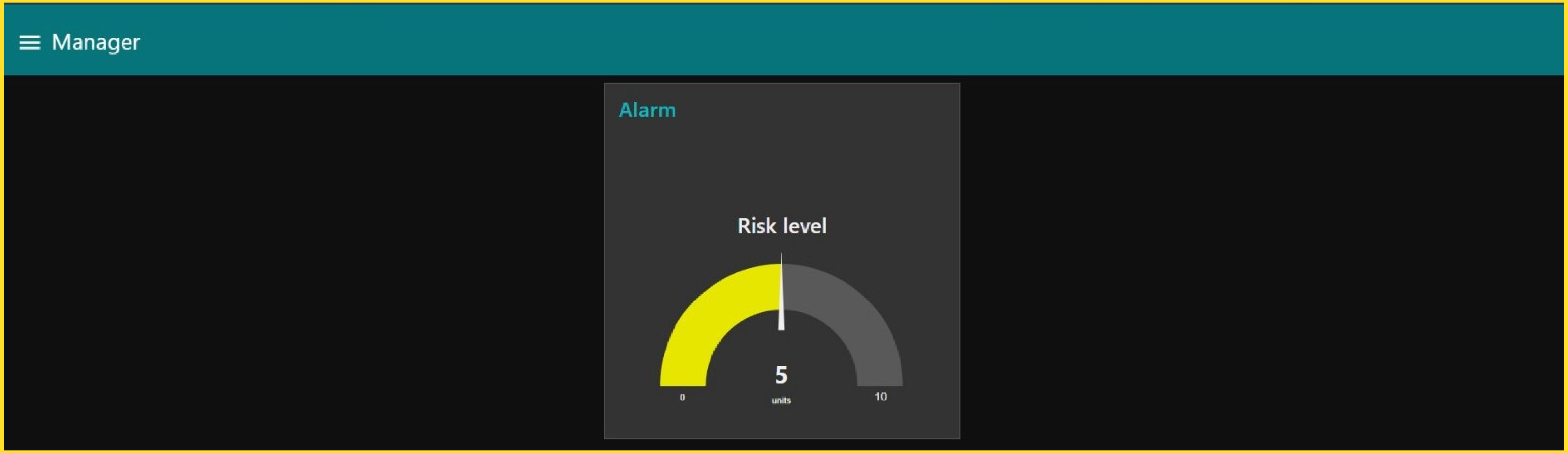
Based on the message sent from the model, the driver status is monitored by his manager and an action is taken in the driver car (Alarm is going on) if he is going to sleep and a gauge is connected for the risk level.

Monitoring inside the driver’s car



You are going to sleep, drink coffee please!!

Monitoring for the driver’s manager



Manager
Alarm
Risk level
5
units
0
10