# SMART GLASSES

## A PROJECT REPORT

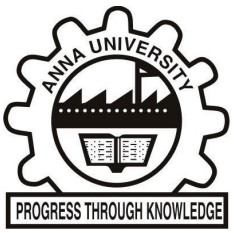
**Submitted by**


| | |
|---|---|
| **ABDUL RAHEEM** | **2019504501** |
| **SANTHOSH S** | **2019504578** |
| **TARUN U** | **2019504597** |
| **VAIKUNTH GURUSWAMY** | **2019504602** |


**in partial fulfilment for the award of the degree**

**IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**





**MADRAS INSTITUTE OF TECHNOLOGY**

**ANNA UNIVERSITY: CHENNAI 600 044**

**NOV 2021**

**EC5561 - Microprocessor and Microcontroller Interfacing Laboratory**

## TABLE OF CONTENTS

**MADRAS INSTITUTE OF TECHNOLOGY**

**ANNA UNIVERSITY**

## BONAFIDE CERTIFICATE

Certified that this project report **SMART GLASSES** is the bonafide work of **ABDUL RAHEEM, SANTHOSH S, TARUN U, VAIKUNTH GURUSWAMY** who carried out the project work under my supervision for the course EC5561 - Microprocessor and Microcontroller Interfacing Laboratory


**SIGNATURE**
Dr. A. Divya
Assistant Professor
Electronics and Communication Engineering
Department of Electronics Engineering
Madras Institute of Technology
Anna University

**SIGNATURE**
Dr. K. Mariammal
Assistant Professor(Sr. Gr.)
Electronics and Communication Engineering
Department of Electronics Engineering
Madras Institute of Technology
Anna University

# 1.ABSTRACT

Among recent inventions,smart glass is one of the wearable devices typically referred to as switchable glass that is capable of handling a wide range of computing activities that an ordinary human cannot do. A survey on smart glass applications was made and various possible new applications were explored. Unlike the possible applications,numerous challenges faced by the smart glasses were explored. In this project, a smart glass that is capable of performing various tasks with voice and video feed is made using suitable algorithms. This project report describes the entire process of its creation from hardware requirements and implementation, it will be compiled and simulated through a Windows application – VS Code in python language which will be connected through the ESP-32 Camera Module programmed in Arduino IDE. This project implements object and text detections with the help of Image Processing which is implemented via Python code and the respective movement in real-time via hardware coding using Arduino.

A total set of 14 relevant publications was identified to help arrive at a definition and characteristics of smart glasses. Therefore, an adapted definition of smart glasses is developed based on the existing original rationale of ubiquitous computing and taking the current state-of-the-art knowledge into account.

# 2. INTRODUCTION

Smart glasses are computing devices worn in front of the eyes. Evidently their displays move with the users head, which leads to the users seeing the display independently of his or her position and orientation. Therefore smart glasses or lenses are the only devices which can alter or enhance the wearer’s vision no matter where he/she is physically located and where he/she looks.

Like other smart devices, smart glasses will often also have a camera. Significant differences to other camera devices are that the pictures or videos are taken from the users point of view, there is no need for the user to hold the device in his hands and the vision of the user is not occluded. This cam-era can see what the wearer sees at any time. In combination with eye tracking technology the devices can determine ex- actly what the wearer is looking at. This allows the device to get crucial information about the user's interests, activities, surroundings and occupation. Those fundamental differences to other computing devices are what makes smart glasses unique and interesting. They enable new applications which couldn’t be as easily realized with other devices.

## Literature survey


1. Nallapaneni Manoj kumar, Neeraj kumar Singh, Vijay Kumar peddiny,
   "Wearable Smart Glass : Features, Applications, Current Progress and Challenges", IEEE-2018.

2. Lik - Hang Lee, Pan Hui , " Interaction methods for smart glasses",IEEE 2014.

3. Anna Syberfeldt , Oscar Danielsson , Patrik Gustavsson, " Augmented Reality Smart Glasses in the Smart Factory: Product Evaluation Guidelines and Review of Available Products", IEEE 2017.

4. Nanoka Sumi, Vasily Moshnyaga , " A novel face recognition for smart glasses" , IEEE 2016.

5. Nikitha Kommera, Faisal Kaleem, Syed Mubashir Shah Harooni, " Smart augmented reality glasses in cybersecurity and forensic education", IEEE 2016.
6. Oscar Danielsson, Magnus Holm, Anna Syberfeldt,"Evaluation Framework for Augmented Reality Smart Glasses as Assembly Operator Support: Case study of Tool Implementation" , IEEE 2019.

7. P.Selvi Ragendran, Padmaveni krishnan, D.John Aravind, " Design and Implementation of Voice Assisted Smart Glasses for Visually Impaired People using Google Vision API"2016.

# 3. HARDWARE

## 3.1.Esp32:

The ESP32-CAM is a very small camera module with the ESP32-S chip. Besides the OV2640 camera and several GPIOs to connect peripherals, it also features a microSD card slot that can be useful to store images taken with the camera or to store files to serve clients.

The ESP32-CAM doesn't come with a USB connector, so you need a USB to TTL UART programmer like CP2102 to upload code through the U0R and U0T pins (serial pins).

- The smallest 802.11b/g/n Wi-Fi BT SoC module
- Low power 32-bit CPU, can also serve the application processor
- Up to 160MHz clock speed, summary computing power up to 600 DMIPS
- Built-in 520 KB SRAM, external 4MB SRAM
- Supports UART/SPI/I2C/PWM/ADC/DAC
- Support OV2640 and OV7670 cameras, built-in flash lamp
- Support image WiFI upload
- Support TF card
- Supports multiple sleep modes

ESP32-CAM Pinout

The following figure shows the ESP32-CAM pinout (AI-Thinker module). There are three GND pins and two pins for power: either 3.3V or 5V.

GPIO 1 and GPIO 3 are serial pins. You need these pins to upload code to your board. Additionally, GPIO 0 also plays an important role, since it determines whether the ESP32 is in flashing mode or not. When GPIO 0 is connected to GND, the ESP32 is in flashing mode.

The following pins are internally connected to the microSD card reader:

- GPIO 14: CLK
- GPIO 15: CMD
- GPIO 2: Data 0
- GPIO 4: Data 1 (also connected to the on-board LED)
- GPIO 12: Data 2
- GPIO 13: Data 3

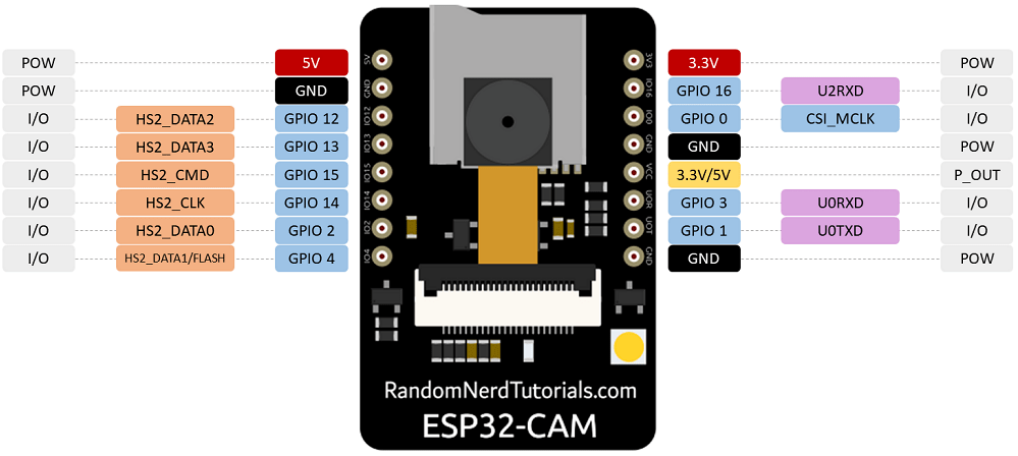
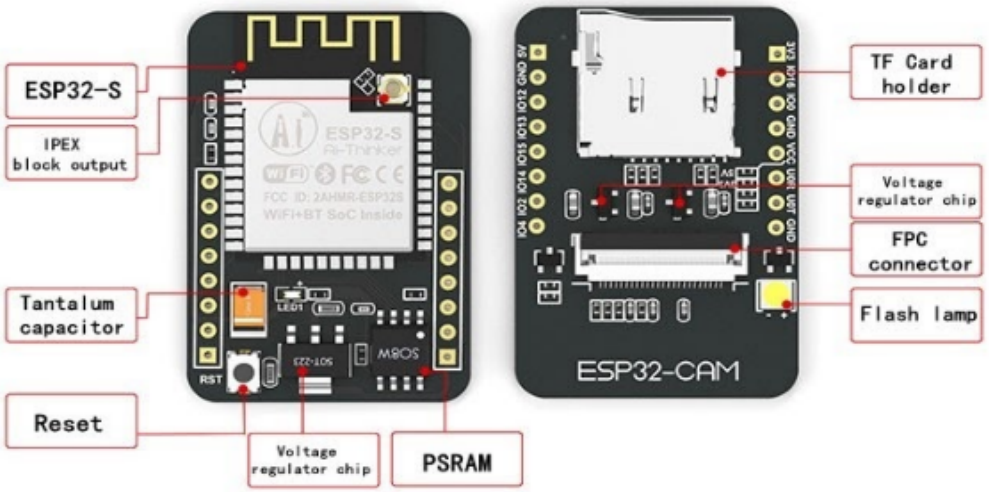



FIGURE 3.1.1: ESP 32 PIN DETAILS




sst

FIGURE 3.1.2: ESP 32 Camera Module

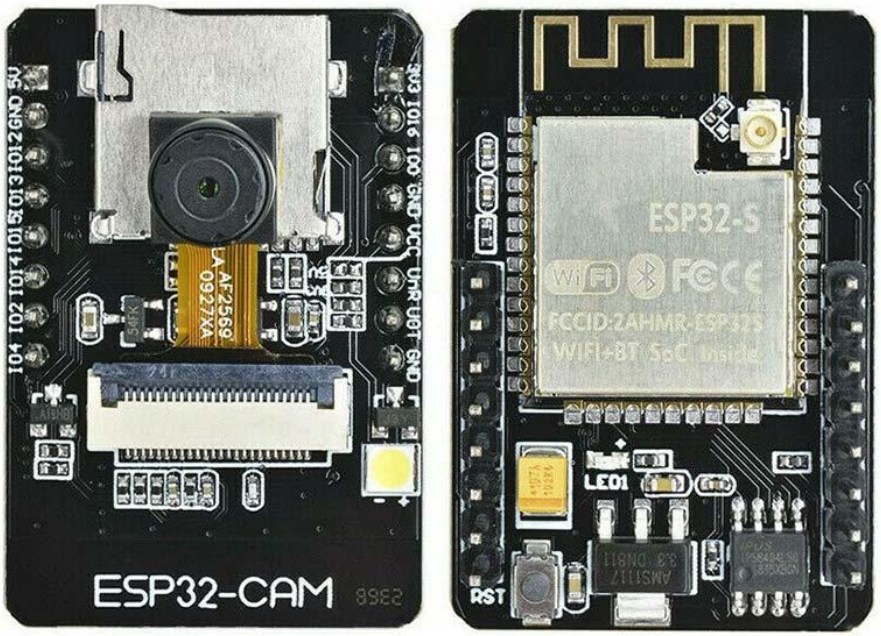



FIGURE 3.1.3: ESP32 Camera Module (Front and Back)

### 3.2.MB102 Breadboard Power Supply

MB102 Breadboard Power Supply module is one of the essential and low-cost components in the electronics labs. It powers the circuits and is also used for testing purposes. The small compact module is power efficient and can be operated using an input voltage range of 6.5 Volts to 12 Volts. The module has two voltage regulators which output 3.3 Volts and 5 Volts. Mb-102 also has an onboard capacitor for noise suppression and smoothing the input voltage.

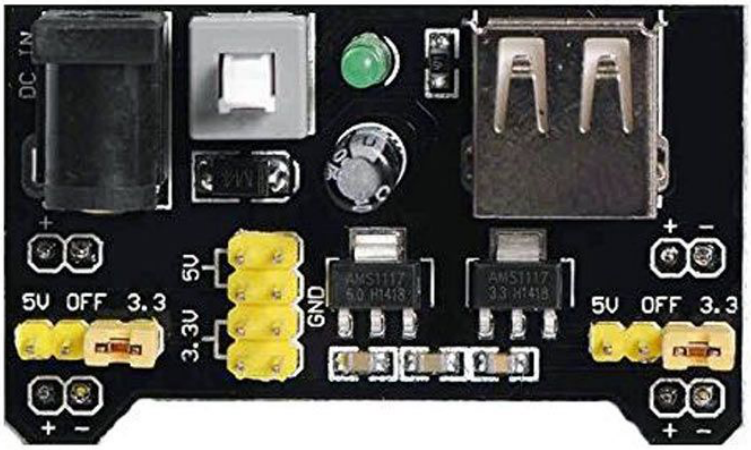



FIGURE 3.2.1: MB102 Power Supply Module

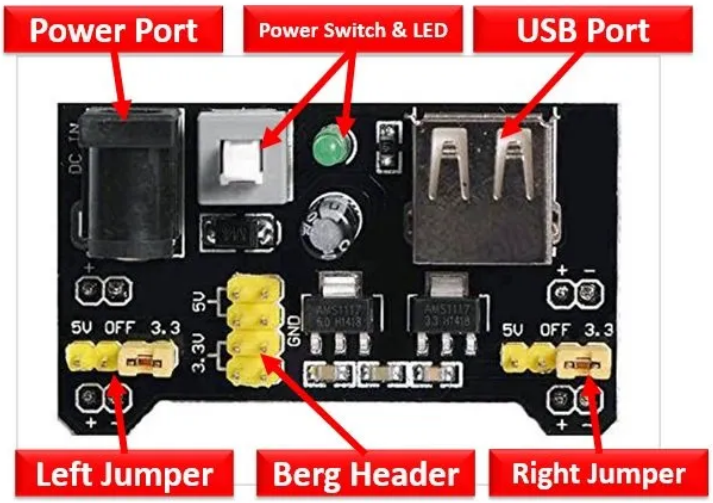



FIGURE 3.2.2: MB102 Power Supply Module Port Details

The breadboard power supply module consists of:

Power Port & USB Port: The DC power port and USB-A connector are provided to the module to power it up.

Power Switch & LED: A switch is embedded to provide extra control along with an LED to indicate the energizing of the module.

Jumpers: The mb102 breadboard supply module is capable of giving out 3.3 volts or 5 volts to breadboard rails. They can be operated individually.

Berg Headers: The berg headers can be used to output power to other devices as well.

Features and Specifications

- Operating Input voltage: 6.5 Volts – 12 Volts
- Output voltage: 3.3 Volts or 5 Volts
- Maximum Output Current: < 700 mA
- Module Dimensions: 5.3cm x 3.5cm
- The module has an on/off switch to control the external input switch.
- Along with the DC port, the Breadboard module has a USB port.
- The USB port provides input to the module to output power to the circuits.
- The module has selectable power rails that can be controlled independently.
- The module can switch between output voltages i.e 3.3V and 5V.
- Plug the BBPS module directly into the breadboard.
- For easiness, the BBPS module also has two pairs of onboard 3.3V and 5V DC output berg headers.

## 3.5.CP2102

The CP2102 is a highly-integrated USB-to-UART Bridge Controller providing a simple solution for updating RS-232 designs to USB using a minimum of components and PCB space. The CP2102 includes a USB 2.0 full-speed function controller, USB transceiver, oscillator, EEPROM, and asynchronous serial data bus (UART) with full modem control signals in a compact 5 x 5 mm MLP-28 package. No other external USB components are required.

USB Function Controller and Transceiver

The Universal Serial Bus function controller in the CP2102 is a USB 2.0 compliant full-speed device with an integrated transceiver and on-chip matching and pull-up resistors. The USB function controller manages all data transfers between the USB and the UART as well as command requests generated by the USB host controller and commands for controlling the function of the UART.

Asynchronous Serial Data Bus (UART) Interface

The CP2102 UART interface consists of the TX (transmit) and RX (receive) data signals as well as the RTS, CTS, DSR, DTR, DCD, and RI control signals. The UART supports RTS/CTS, DSR/DTR, and XOn/XOff handshaking.

The UART is programmable to support a variety of data formats and baud rates. The data format and baud rate programmed into the UART are set during COM port configuration on the PC.

Voltage Regulator

The CP2102 includes an on-chip 5 to 3 V voltage regulator. This allows the CP2102 to be configured as either a USB bus-powered device or a USB self-powered device. These configurations are shown in Figure 7 and Figure 8. When enabled, the 3 V voltage regulator output appears on the VDD pin and can be used to power external 3 V devices. See Table 8 for the voltage regulator electrical characteristics.

Alternatively, if 3 V power is supplied to the VDD pin, the CP2102 can function as a USB self-powered device with the voltage regulator disabled. For this configuration, it is recommended that the REGIN input be tied to the 3 V net to disable the voltage regulator. This configuration is shown in Figure 9. The USB max power and power attributes descriptor must match the device power usage and configuration. See application note “AN144: CP210x Customization Guide” for information on how to customize USB descriptors for the CP2102.

FIGURE 3.5.1: CP2102 Module

## 4.Software

The software code used here is Python. Python is a general purpose interpreted high level language.It gives access to a wide range of libraries and helps easier readability and user friendly coding environment.The features that have been incorporated to this project using Python code are implemented via the following functionalities.

### 4.1 Python Functionalities

- Web Scraping:

Web Scraping also known as Web Data Extraction or Web Harvesting is the use of accessing the data from the World wide Web through the HyperText Transfer Protocol. In this way the functionality of interest of any webpage is directly accessed by the Python Code. Web scraping has been around since the early days of the World Wide Web, but scraping modern sites that heavily rely on new technologies is anything but straightforward. Libraries used for Web Scraping
As we know, Python has various applications and there are different libraries for different purposes. In our further demonstration, we will be using the following libraries:

- Selenium: Selenium is a web testing library. It is used to automate browser activities.
- BeautifulSoup: Beautiful Soup is a Python package for parsing HTML and XML documents. It creates parse trees that are helpful to extract the data easily.
- Pandas: Pandas is a library used for data manipulation and analysis. It is used to extract the data and store it in the desired format.
-

Web scraping, although done manually, is usually referred to be done using a web crawler. Web crawlers are usually updated with the URL from

which they extract all the data from which the required functions are accessed using the specific class.

- Image Processing:

  Image processing in Python is done using the OpenCV module. It is an exclusive module available for processing images. OpenCV is an open source library which helps in facilitating computer vision and is employed in Artificial Intelligence, Machine Learning, etc.

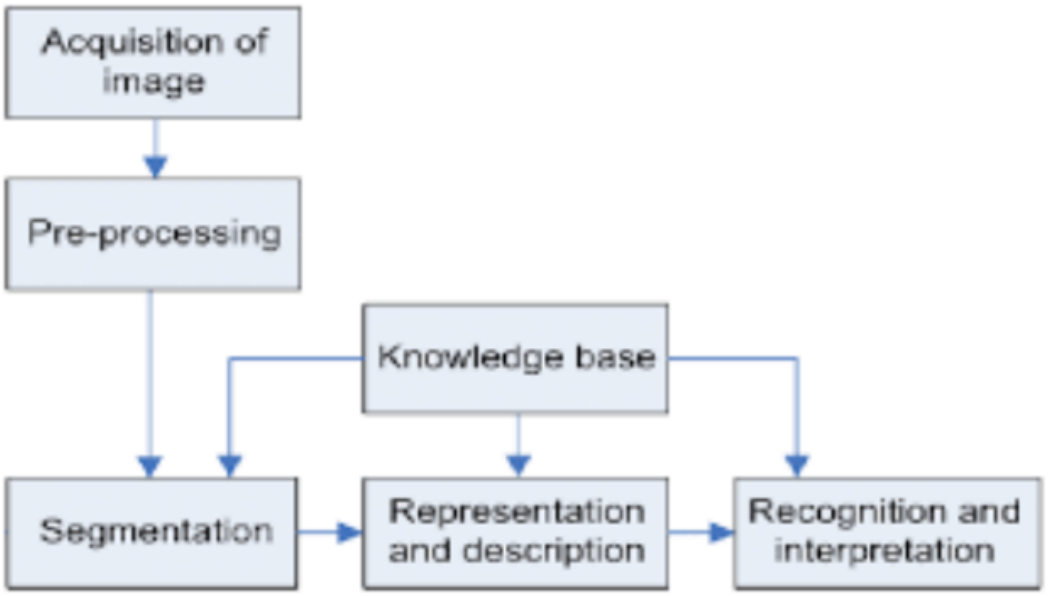



Figure 4.1 image proccesing

Digital image processing is concerned with processing of an image. Image processing is a method to perform operations on images like enhancing images, extracting text from image,detecting edge of image and many other operations. In digital image processing we take an image and convert that image in different forms. Like if we take a color image we can convert it into a grey image. In this both the input and output is an image. Usually an Image Processing system includes treating images as two

dimensional signals while applying already set signal processing methods to them.

Today, it is a rapidly growing technology. It forms a core research area within engineering and computer science disciplines too. Image processing has its wide applications in robotics, machine learning, neural networking, signal processing, medical field, graphics and animations and in many other fields.

Machines identify the objects based on the intensity of the colour which is interpreted as a set of numbers and the so generated values are used for processing the image so captured. The system usually interprets the image in two formats,

- Grayscale:

  Here the captured image is stored as a collection of black and white pixels whose intensity of colour changes with respect to the intensity of the colour.The weakest intensity is considered to be black and the highest intensity is considered to be white.

- RBG:

  Any visible colour can be expressed as a combination of the three basic colours Red, Blue and Green.In case of Python each individual pixel is interpreted as a tuple of three elements consisting of the three values which ranges from 0 to 255 for each of the values which decides the shade and intensity of the colour.

- GUI Interfacing:

  Python supports the creation of Graphical User Interface using various methods. Pyqt5 is one of the conventional methods used for creating the GUI in an user-friendly manner.

  Qt Designer is used in addition to the above mentioned module. The mentioned application used to generate the general setup of the user interface main window like setting the position of a text box, labels, buttons,etc. The functions, features and the output

of these individual elements of the main window are to be changed or given using the Python Code.

The Qt designer files are stored in the ".ui" file format which can be converted to the ".py" file format by using the command prompt of the respective OS. The command for converting the file is,

```
pyuic5 -x  Filename.ui -o Filename.py
```

The -x converts the required ".ui" file as an executable file and is converted to a python code file which can be operated using the required interpreter.

- File Accessing:

  Python accessing data files in order to obtain the required data for executing the commands based on the instructions of the user.In this project there are three types of files being accessed,

  - CSV:

    A CSV or Comma Separated Values  data file is usually a text or excel sheet file which stores the specifications of a specific variable of interest in the form of comma separated values.This type of files are easily processed using the Python Pandas Module which are exclusively available for processing CSV data files and are made easy and more accessible.

  - CAFFEMODEL:

    A CAFFEMODEL file is a type of format which is an exclusive machine learning model. It contains an image classification or segmentation model that has been trained using the Caffe. Caffe is a deep learning framework. The CAFFEMODEL serves as the base data for a machine for classifying the data and returning the desired output.

- Prototxt:

  Prototxt files are the base type files used in deploying the CAFFEMODEL file. The CAFFEMODEL files are initially created and saved using the Prototxt type file and then after training and refining are saved as a CAFFEMODEL.

  This type of file contains a list of neural network layers, each layer with its own set of parameters like name type input dimensions and output dimensions, and specifications of the connection between layers.

- Audio Detection:

  Python supports speech recognition and audio detection by initially detecting the audio signal generated during speech and then the signal is sampled and processed. Then the processed samples are compared with the speech recognition library data in order to recognise what the user wishes to convey in text form.

## 4.2 Functionality Implementation

- Chat Bot:

  In this project there is a chat bot which is being added to the interface which interprets the voice command being given by the user and responds accordingly based on the instruction data set given to it in real time.

  It has been built using the Pyqt5 module for building the User Interface using Python code,and HTML which makes the whole session with the bot more user friendly and the bot has access to a huge data set of values for responding to the user and accessing the data of the other functionalities being embedded within this project.

- Maps:

  This Project comes with an inbuilt navigator which helps in finding routes to the desired destination from any location in real time. THis functionality of the project is obtained via Google Maps.

  When you try to share your live location in a WhatsApp chat - by tapping the Attach icon on Android, or the Plus icon on the iPhone - you will be prompted to select the duration of time for which the share will be active. You can only choose from one of the preset options - 15 minutes, 1 hour (default), and 8 hours - and, optionally, add a comment. Khan said that these durations had been picked based on the common scenarios in which people are most likely to use WhatsApp live location sharing. Users can still choose to share a one-time, static location if they prefer.

  You can manually stop sharing live location data within a chat at any given time. A new option in your Account Privacy settings will give you an overview of all WhatsApp chats you are sharing your live location in at the time.

  Live location shares will show up in WhatsApp chats as thumbnails that show the initial location of the user and a ‘Live until’ time. Tapping ‘View live location’ will take you to a map view where you can see the current location of all the users who are presently sharing their location in that chat. You can switch to a Satellite/ Terrain view of the map if you desire, and even see live traffic data as an overlay, which sounds pretty convenient if you are waiting for someone.

  Below the map you will see the names of the same users in a list view, with a timestamp that indicates the freshness of their location data. You can tap a name on the list or tap a profile picture on the map to zoom in to the location of a particular user where you will also see the margin of error (for example “Accurate to 50 metres”) for their location. You can tap the little (i) button next to the

user to bring a pop-up that lets you communicate with them using WhatsApp in different ways.

In a WhatsApp group, you can bring up the group info and see all users that are sharing their location with the group at that point in time. This will be particularly useful in busier groups, as you won't need to scroll up and try to find the message where the user originally shared their location.

"We hope that it provides utility in those scenarios where people are coordinating and meeting up in the real world. We give users control so they can stop sharing at any time and sharing a live location is just as secure as sending any other communications over WhatsApp in that it is end-to-end encrypted," Khan said.

Here we use web scraping to access the required qualities of the Google Maps and deliver the required results. Despite being able to access the WWW for additive information the coordinates of the locations in the maps has to be fed beforehand to generate the desired path in the output. With this constraint in mind the Python code is initially given access to a data file which contains the coordinates of almost every location of interest.

Once the initial and final locations are fed to the program the coordinates are accessed and they serve as the data for processing.

- Object Detection:

  Object recognition is to describe a collection of related computer vision tasks that involve activities like identifying objects in digital photographs. Image classification involves activities such as predicting the class of one object in an image. Object localization refers to identifying the location of one or more objects in an image and drawing a bounding box around their extent. Object detection does the work of combines these two tasks and localizes and classifies one or more objects

in an image.

In this functionality both object and text are being detected using the OpenCV module.In the process of Text detection we have the following steps involved,

1. First, the image captured using the by accessing the ESP32 camera module.
2. The text is detected based on the data set given beforehand to the program .
3. The language of the displayed text is detected.
4. Then the text is cropped out from the rest of the image.
5. Then the appropriate synonym of the displayed text is displayed on top of the image itself in english.

In the process of object detection we have the following steps involved,

1. First, the image captured using the by accessing the esp 32 camera module.
2. The objects are detected based on the data set given beforehand to the program .
3. Then the object is cropped out from the rest of the image.
4. Then the name of the object is displayed on top of the cropped box image itself in english.

- Music Player:

  The music player plays the required music by accessing the online database, in this case YouTube and acquires the required audio data file.
  Then the file is played by the system using the VLC Media player.

  This AI Assistant is getting a new feature that identifies any song playing in the background, similar to how google assistant works. The feature was originally released a month ago, but was limited to Google's Pixel devices. Now, the feature is being rolled out for Android devices supporting Google Assistant.

The user will feed the instruction via voice command and the respected music will be played from online.

## 4.3 ARDUINO IDE

**Arduino IDE** is open-source software, designed by Arduino.cc and mainly used for writing, compiling & uploading code to all Arduino Modules and other modules too. It is based on Processing. Arduino code is written in C, C++ with an addition of special methods and functions and is based on wiring.

Arduino IDE has 2 parts:

i) Editor for writing the required code.
ii) Compiler for compiling and uploading the code.

Arduino IDE environment has 3 sections:

i) Menu and toolbar containing options like Verify, Upload, File, Sketch, Serial monitor, etc.
ii) Text editor where the text is written.
iii) Output pane where compilation status of running code and errors are shown and status bar showing the board and the port used

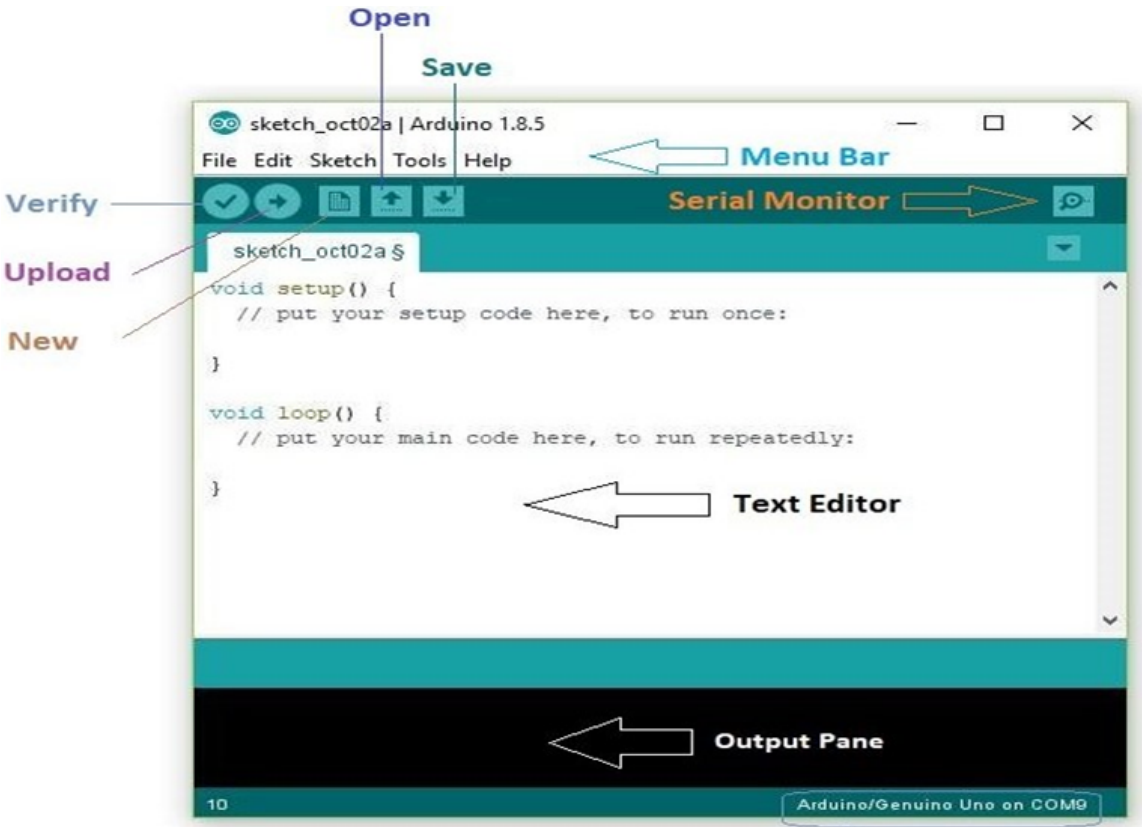



Figure 4.2 arduino software

The main code, also known as a sketch, created on the IDE platform will ultimately generate a Hex File which is then transferred and uploaded in the controller on the board. Arduino files are saved as .uno file extensions. Arduino IDE is preferred because it is cross-platform (can run on Windows, macOS, Linux), is easy to use for beginners, and flexible for advanced programmers too as advanced programming can be done too and can be extended too, can be extended to boards other than Arduino boards, has lots of libraries which are very helpful and has a serial monitor which helps to easily visualize the error.

Arduino IDE is used in our project to upload code into the ESP32 camera module. The ESP32 camera module can be added by going to File> Preferences and then entering https://dl.espressif.com/dl/package_esp32_index.json into the "Additional Board Manager URLs" field and then, clicking "OK". Now, the board will be added to IDE and can be downloaded using Tools > Board > Boards

Manager and pressing the install button on the “ESP32 by Espressif Systems”. After being installed, again go to Tools > Board to select the required ESP32 board. Additional libraries can be downloaded if needed and added to the IDE.

Coding is done using Arduino code with some differences for the ESP32 camera module. Code is verified and uploaded into the ESP32 camera module and if any error is detected, it is then corrected. The serial monitor is useful for debugging to find the errors in the code. It is found on the right-hand side of the toolbar and can be opened only when the board is connected to the system. Before uploading, the correct port where the device is inserted is to be mentioned or errors will arise. Input can be given to and output can be taken from the serial monitor which acts as an interface between the board and user to find the problems in the code and debug them.

# 5.OUTPUT IMAGES

- **Object detection:**

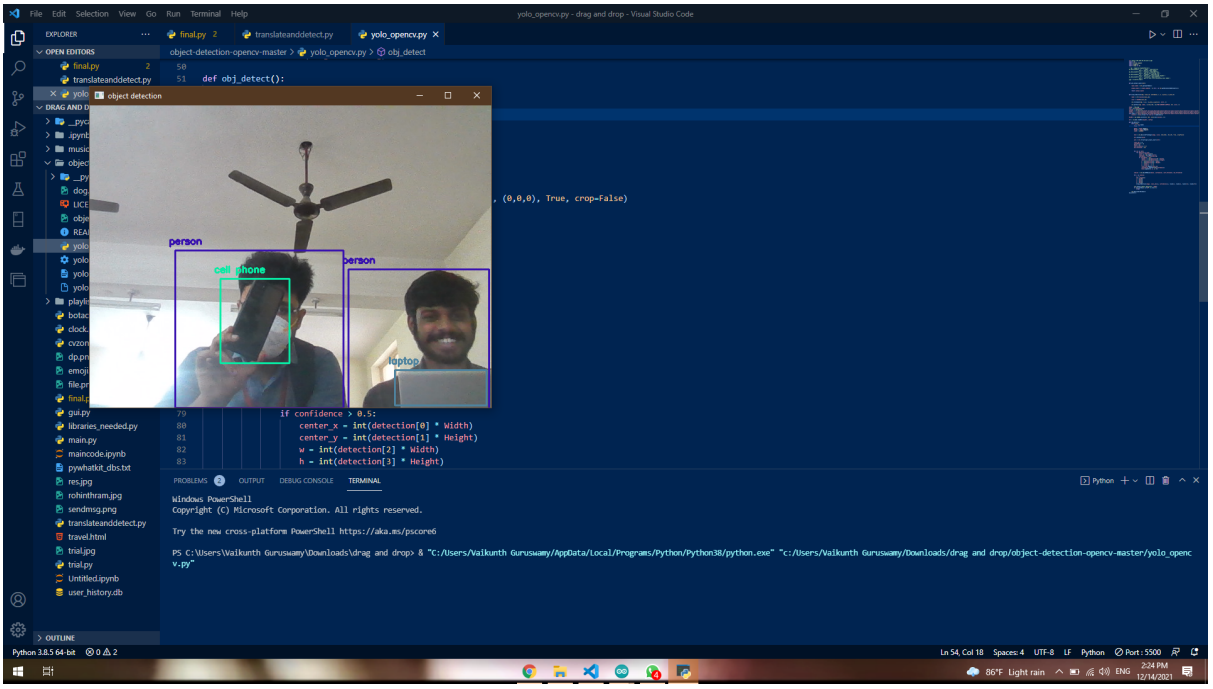



Figure 5.1 object detection

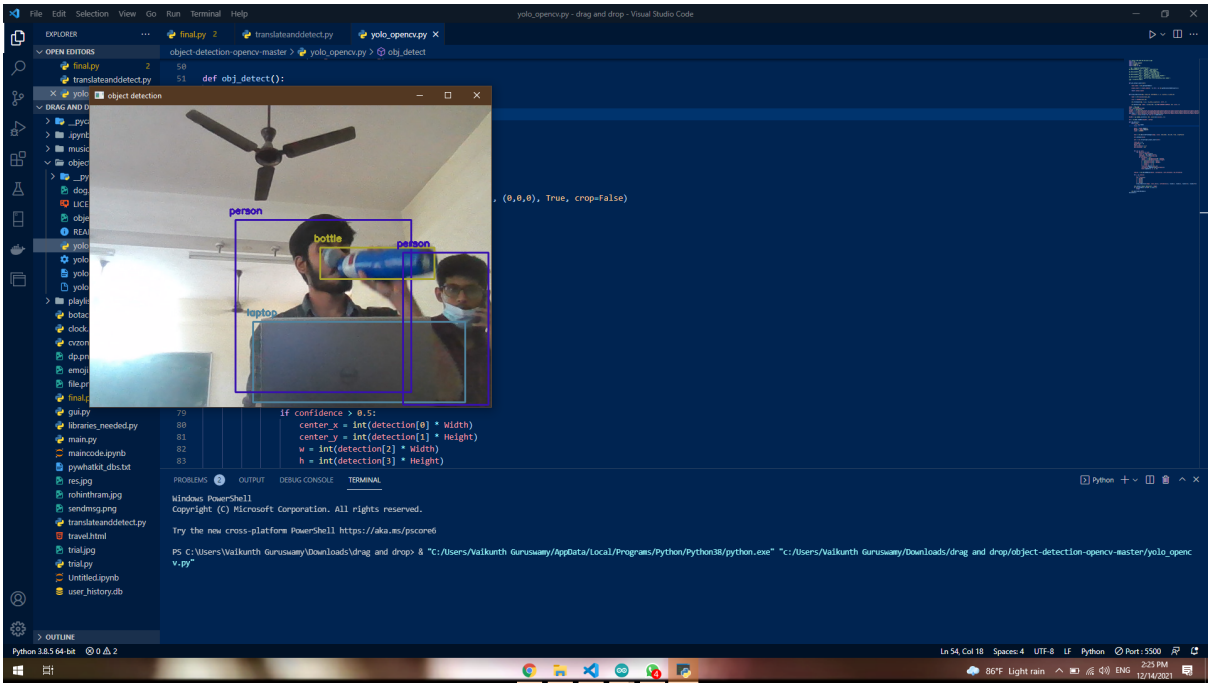



Figure 5.2 object detection

- Music player

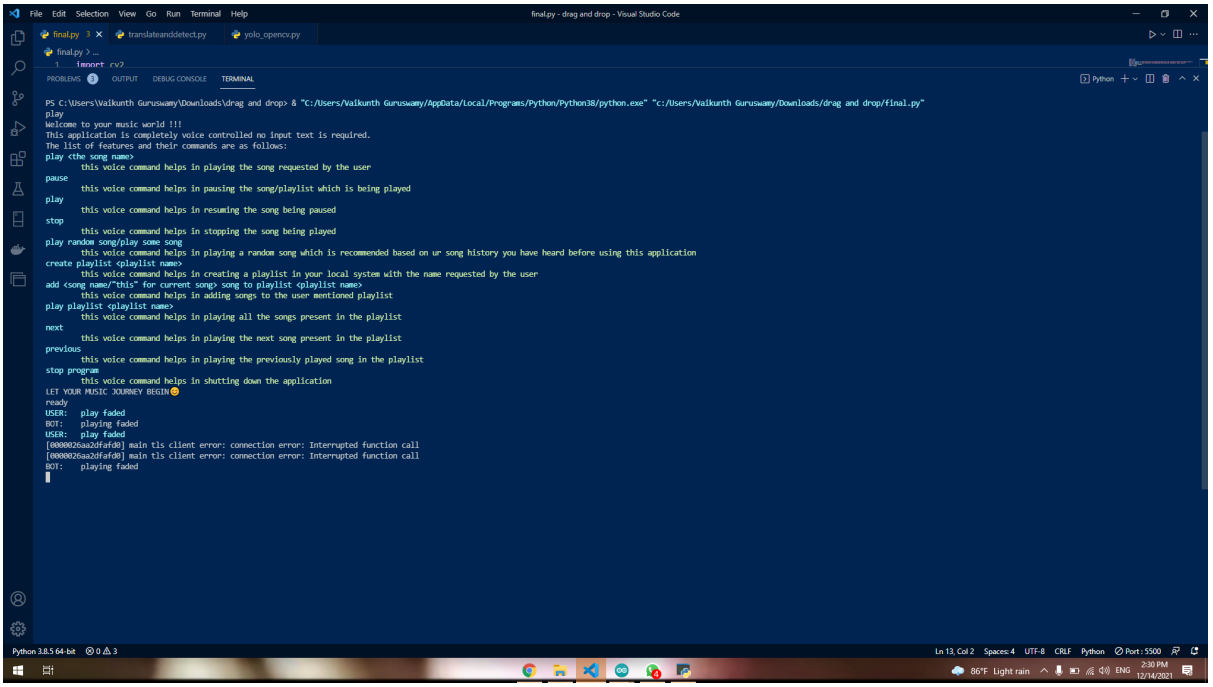



Figure 5.3 music player

- Map integration

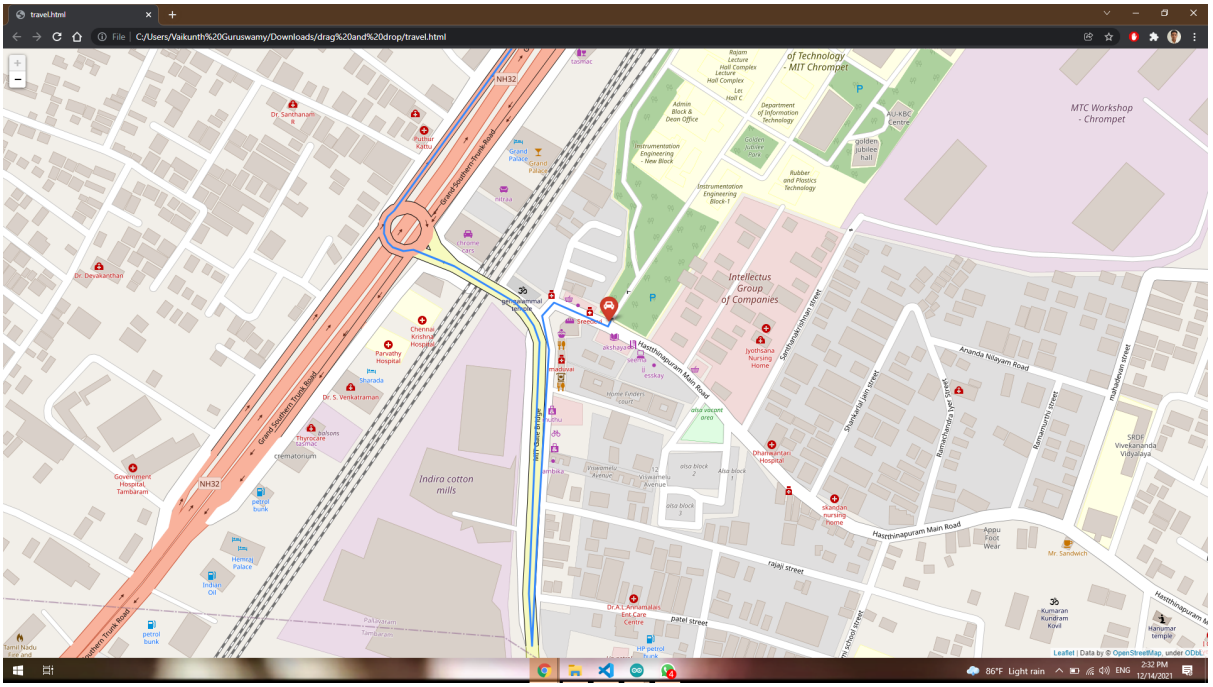



Figure 5.4 map integration

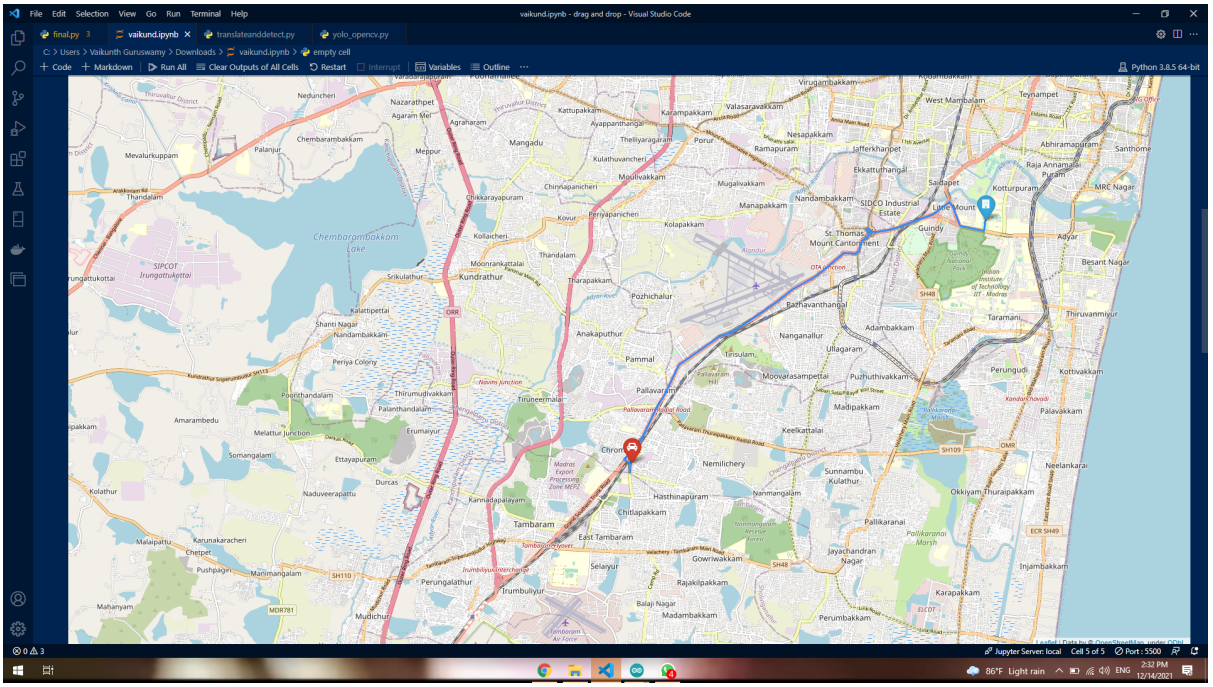
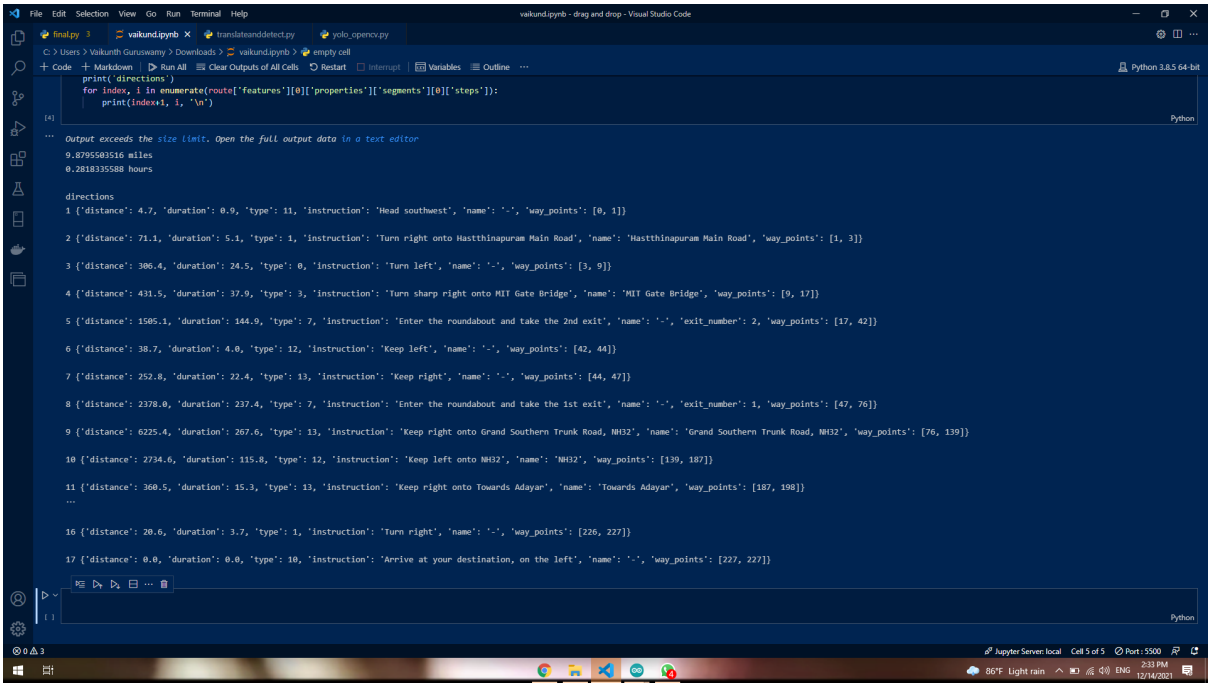


Figure 5.5 map integration



```
print('directions')
for index, i in enumerate(route['features'][0]['properties']['segments'][0]['steps']):
    print(index+1, i, '\n')
```

```
Output exceeds the size limit. Open the full output data in a text editor
9.8795503516 miles
0.2818335588 hours

directions
1 {'distance': 4.7, 'duration': 0.9, 'type': 11, 'instruction': 'Head southwest', 'name': '-', 'way_points': [0, 1]}

2 {'distance': 71.1, 'duration': 5.1, 'type': 1, 'instruction': 'Turn right onto Hastthinapuram Main Road', 'name': 'Hastthinapuram Main Road', 'way_points': [1, 3]}

3 {'distance': 306.4, 'duration': 24.5, 'type': 0, 'instruction': 'Turn left', 'name': '-', 'way_points': [3, 9]}

4 {'distance': 431.5, 'duration': 37.9, 'type': 3, 'instruction': 'Turn sharp right onto MIT Gate Bridge', 'name': 'MIT Gate Bridge', 'way_points': [9, 17]}

5 {'distance': 1505.1, 'duration': 144.9, 'type': 7, 'instruction': 'Enter the roundabout and take the 2nd exit', 'name': '-', 'exit_number': 2, 'way_points': [17, 42]}

6 {'distance': 38.7, 'duration': 4.0, 'type': 12, 'instruction': 'Keep left', 'name': '-', 'way_points': [42, 44]}

7 {'distance': 252.8, 'duration': 22.4, 'type': 13, 'instruction': 'Keep right', 'name': '-', 'way_points': [44, 47]}

8 {'distance': 2378.0, 'duration': 237.4, 'type': 7, 'instruction': 'Enter the roundabout and take the 1st exit', 'name': '-', 'exit_number': 1, 'way_points': [47, 76]}

9 {'distance': 6225.4, 'duration': 267.6, 'type': 13, 'instruction': 'Keep right onto Grand Southern Trunk Road, NH32', 'name': 'Grand Southern Trunk Road, NH32', 'way_points': [76, 139]}

10 {'distance': 2734.6, 'duration': 115.8, 'type': 12, 'instruction': 'Keep left onto NH32', 'name': 'NH32', 'way_points': [139, 187]}

11 {'distance': 360.5, 'duration': 15.3, 'type': 13, 'instruction': 'Keep right onto Towards Adayar', 'name': 'Towards Adayar', 'way_points': [187, 198]}
...

16 {'distance': 20.6, 'duration': 3.7, 'type': 1, 'instruction': 'Turn right', 'name': '-', 'way_points': [226, 227]}

17 {'distance': 0.0, 'duration': 0.0, 'type': 10, 'instruction': 'Arrive at your destination, on the left', 'name': '-', 'way_points': [227, 227]}
```

Figure 5.6 map integration

- Chat bot


Figure 5.7 chat bot

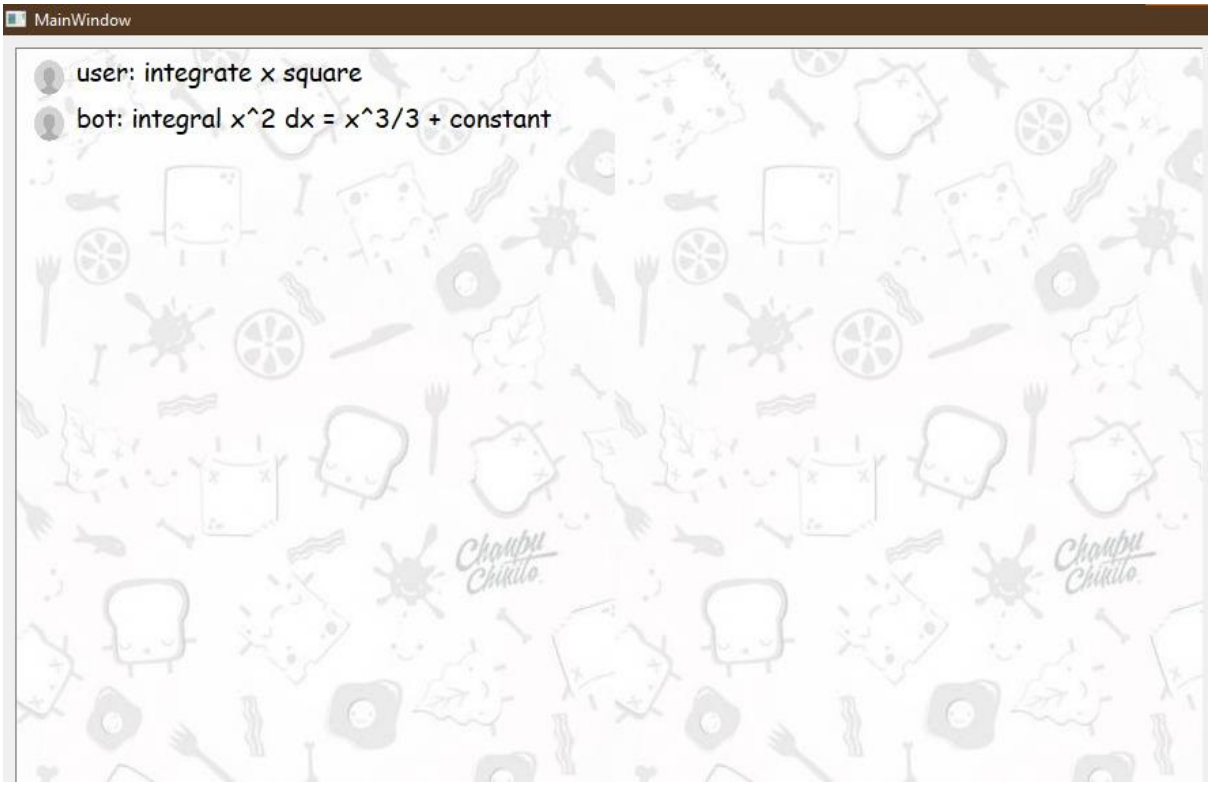



Figure 5.8 chat bot

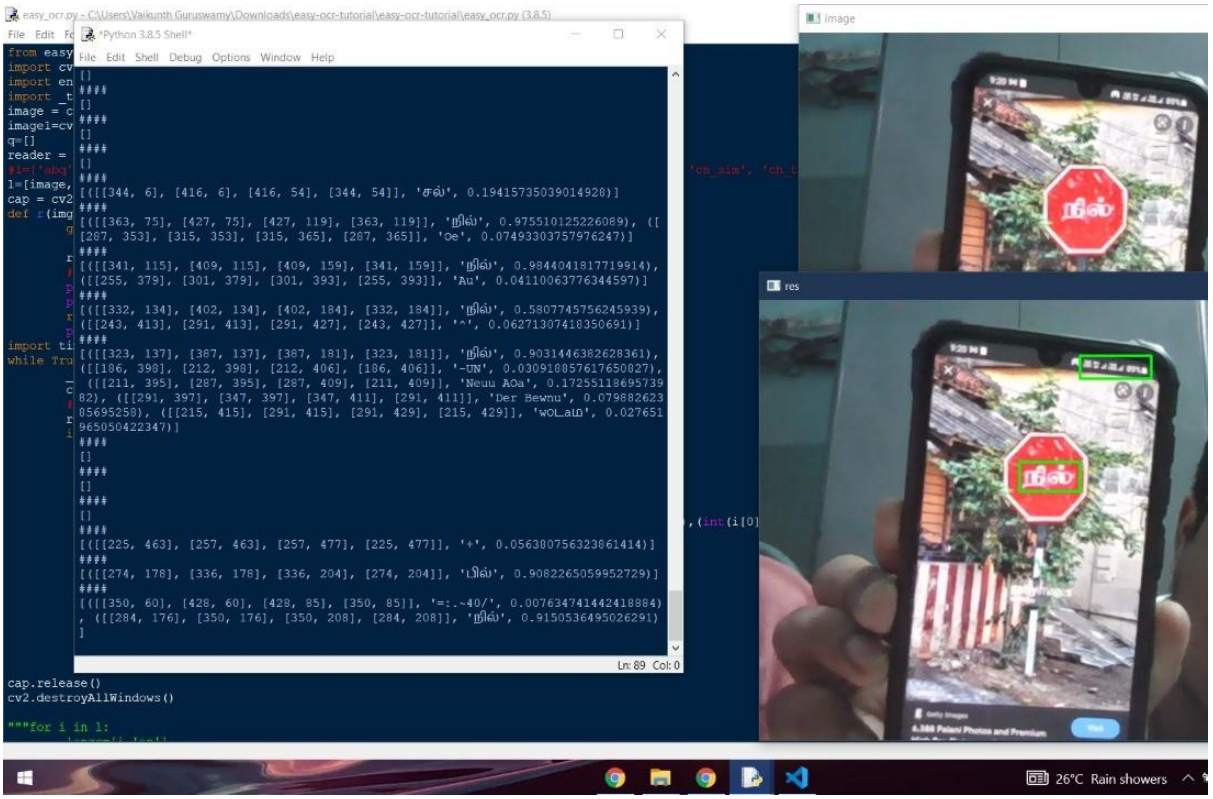



Figure 5.9 text detection and translation

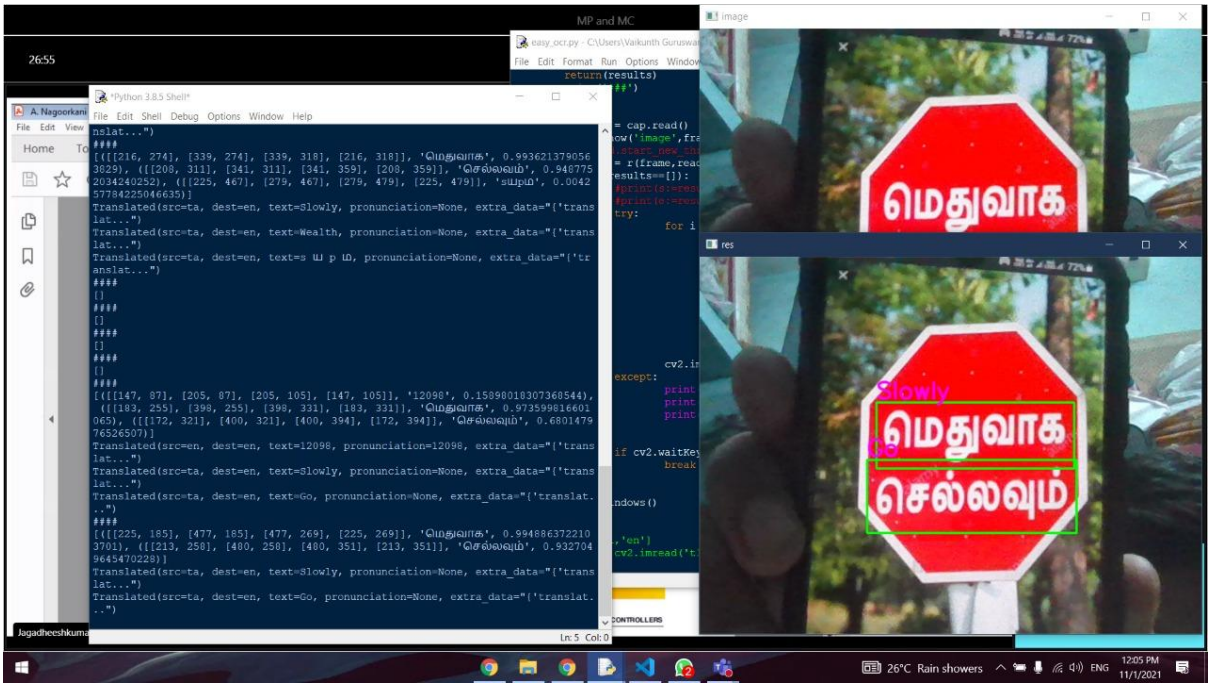



Figure 5.10 text detection and translation

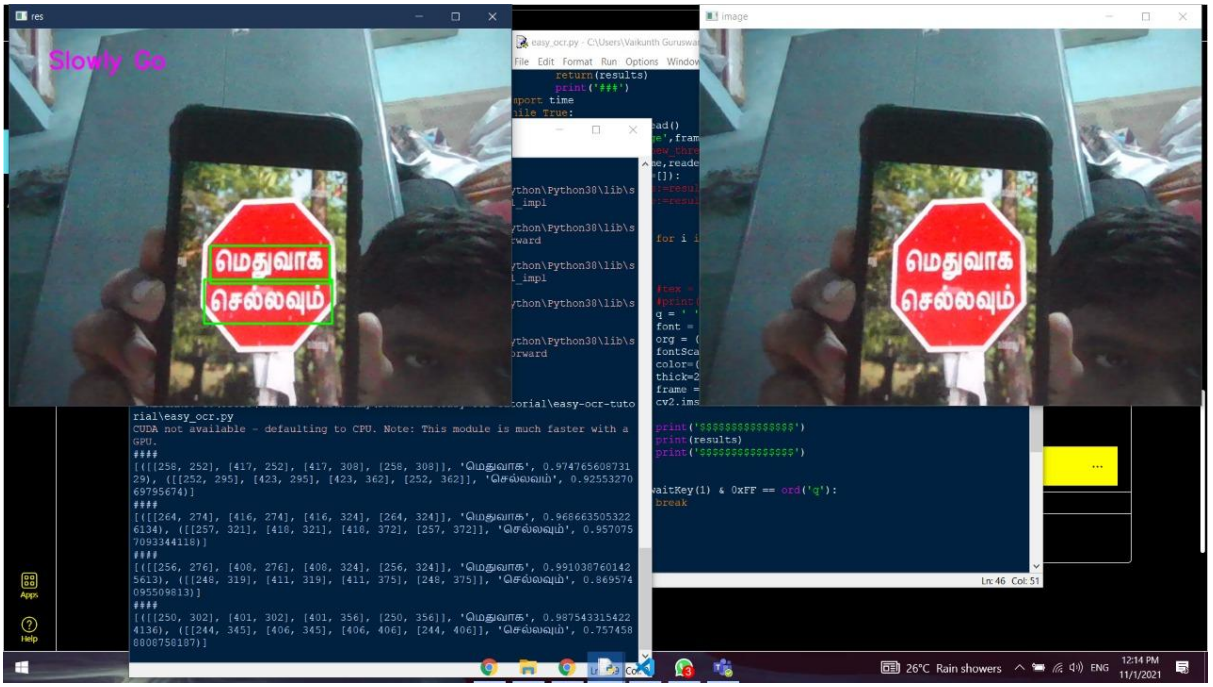



Figure 5.11 text detection and translation

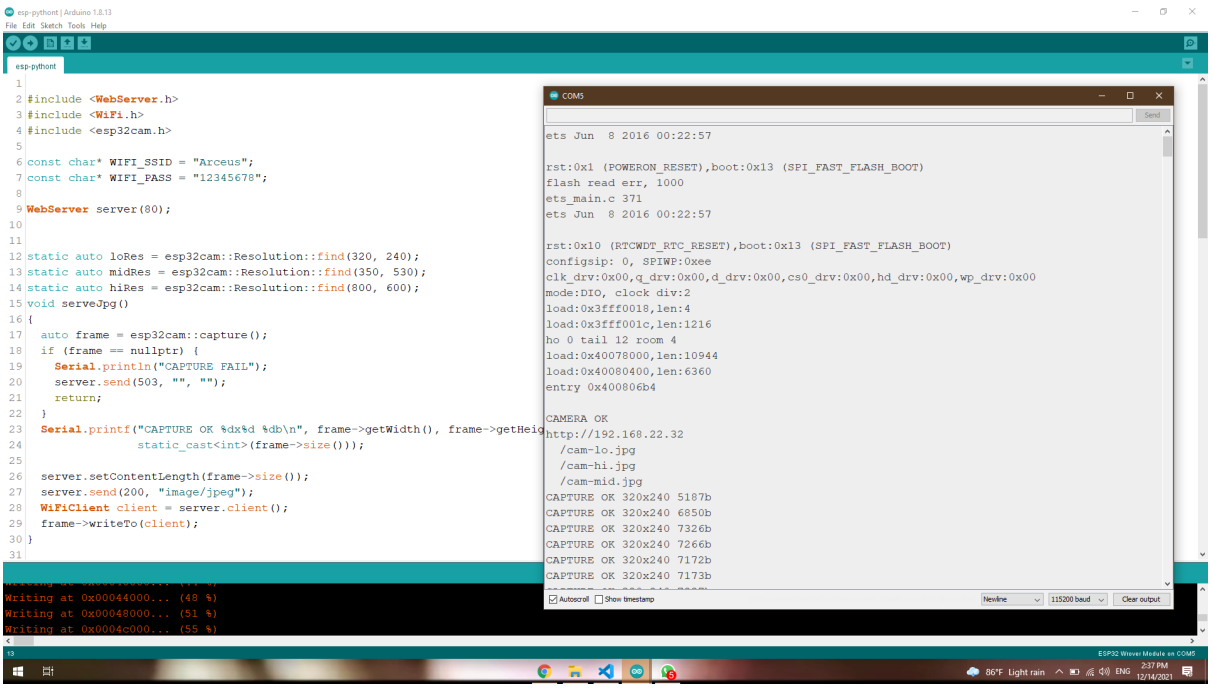



Figure 5.12 arduino code


Figure 5.13 setup image

# 6.CONCLUSION

In conclusion, the wearable smart glass was successfully made and the objectives of the project have been achieved. The aim was to perceive the user's expectation of the visual usage of the device in a different context.

The Smart Glasses was considered as a useful device especially in situations where the user's hands were occupied with other activities and the smart glasses could help the user to multitask and be more productive.

The comprehensive planning and design of the Smart Glasses at an earlier stage made the construction process easy.

Besides, this project provides a good platform for academic learning. We can apply some of the theories learned during class and practice in this project. During the practical time, we are able to understand the actual condition to carry out the project and try to solve all the problems faced. Hence, critical thinking and problem solving are the major issues for us to bring out this Mini Project successfully.

# 7.**FUTURE REFERENCE**

The below mentioned features will be implemented to the smart glass

- E-SIM will be implemented
- T-OLED displayed will be integrated for displaying the output
- Emotion detector

## 8.References


1. Nallapaneni Manoj kumar, Neeraj kumar Singh, Vijay Kumar peddiny,“Wearable Smart Glass : Features, Applications, Current Progress and Challenges”, IEEE-2018.

2. Lik - Hang Lee, Pan Hui , “ Interaction methods for smart glasses”,IEEE 2014.

3. Anna Syberfeldt , Oscar Danielsson , Patrik Gustavsson, “ Augmented Reality Smart Glasses in the Smart Factory: Product Evaluation Guidelines and Review of Available Products”, IEEE 2017.

4. Nanoka Sumi, Vasily Moshnyaga , “ A novel face recognition for smart glasses” , IEEE 2016.

5. Nikitha Kommera, Faisal Kaleem, Syed Mubashir Shah Harooni, “ Smart augmented reality glasses in cybersecurity and forensic education”, IEEE 2016.

6. Oscar Danielsson, Magnus Holm, Anna Syberfeldt,“Evaluation Framework for Augmented Reality Smart Glasses as Assembly Operator Support: Case study of Tool Implementation” , IEEE 2019.

7. P.Selvi Ragendran, Padmaveni krishnan, D.John Aravind, “ Design and Implementation of Voice Assisted Smart Glasses for Visually Impaired People using Google Vision API”2016.