Chapter 3

SYSTEM REQUIREMENTS SPECIFICATION

3.1 General Description of the System

Smart glasses for the visually impaired is an intuitive hardware pilus software combined product which would serve as a helping or assistive device for the visually impaired people or the people who have visual disability. The idea is to design a product by keeping in mind the limitations it would present when being used by a person who us visually disabled.

The developed product will be able to assist the visually impaired people with everyday simple tasks and voice out all the results or the outputs of the processes, to curb this we employ the use text to speech. The idea is the create a product which would voice out the current date and time, voice out the current location, voice out the current weather information with the description, voice out the unread emails, voice out the current top 10 news headlines, voice out the text in an image by using OCR (optical character recognition) technology, voice out the dominant color in an image and also provide obstacle detection using ultrasonic sensors hosted on the product. All these features would be available to the visually impaired person through the push of a button on the handheld remote corresponding to a particular feature. The processing and execution of the various features are performed the raspberry pi processor. All these features are possible by using machine learning models and the python programming language. Python is currently the most widely used multi-purpose, high-level programming language. The biggest strength of Python is huge collection of standard library which can be used for Machine Learning, GUI Applications (like Kivy, Tkinter, PyQt etc.), Web frameworks like Django (used by YouTube, Instagram, Dropbox), Image processing (like OpenCV, Pillow), Web scraping (like Scrapy etc..) Text processing and much more..

3.1.1 Overview of Functional Requirements

The functional requirements give the functionality of the system. For the system to work the mobile data or Wi-Fi with a good internet speed is required. This ensures that the device can work efficiently work with the various API's used in the product. The current weather, current location, current news headlines, text to speech, email voice out features requires a stable internet connection for the efficient results.

3.1.2 Overview of Data Requirements

Data requirements are the consensual agreements or prescribed directives that define the content and structure that consist of high-quality values and data instances. Data requirements can therefore be stated by the several different individuals or groups of individuals. The current date and time feature extracts the required data from the Linux OS, the current location feature extracts the data from the IP to which the system is connected, the current weather feature extracts the data from the open weather API, email voice out feature extracts the data from the email server, the news headlines feature extracts the current news data using the news API, the image to text to speech conversion feature requires an image from the user, the image is taken from the product hosted webcam on the push of a button form the hand held remote, the color detection feature also requires an image from the user, the image is taken from the product hosted webcam on the push of a button form the hand held remote. The ultrasonic sensor analyzes the environment for the obstacles and sets an alarm if an obstacle is too close to the user.

The data for executing a particular feature is taken from the user using the hand held remote. Each button In the remote corresponds to a particular feature on the product. The push a button activates the corresponding feature and the particular feature is executed automatically by the raspberry pi processor.

3.2 Technical Requirements of the System

This technical requirement specifies the requirements of the project. The information on technical design, development and procedures related to the requirements is outlined here. This section talks about the system requirement details, including functional, interface and design requirements.

3.2.1 Hardware Requirements

For the application to be built and used efficiently the hardware components required to run these below mentioned software resources on the computer. It gives a physical computer resource list accompanied by Hardware Compatibility List (HCL).

Raspberry pi processor:

The model of raspberry pi processor used in the project is “Raspberry PI 3 Model B+ ” which has the following specifications:

- SOC: Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC
- CPU: 1.4GHz 64-bit quad-core ARM Cortex-A53 CPU
- RAM: 1GB LPDDR2 SDRAM
- WIFI: Dual-band 802.11ac wireless LAN (2.4GHz and 5 GHz) and Bluetooth 4.2
- Ethernet: Gigabit Ethernet over USB 2.0 (max 300 Mbps). Power-over-Ethernet support (with separate PoE HAT). Improved PXE network and USB mass-storage booting.
- Thermal management: Yes
- Video: Yes – Video Core IV 3D. Full-size HDMI, Audio: Yes
- USB 2.0: 4 ports
- GPIO: 40-pin
- Power: 5V/2.5A DC power input

- Operating system support: Linux and Unix

External Webcam:

The external webcam is used for capturing the images for OCR feature and the dominant color feature. The external webcam used in this project has the following configuration:

- HD video calling (1280 x 720 pixels) with recommended system, Video capture: Up to 1280 x 720 pixels
- Logitech Fluid Crystal Technology, Photos: Up to 3.0 megapixels (software enhanced)
- Built-in mic with noise reduction, Hi-Speed USB 2.0 certified (recommended)
- Universal clip fits laptops, LCD or CRT monitors
- Built-in microphone that reduces background noise.

Compatibility

Basic Requirements

- 1 GHz
- 512 MB RAM or more
- 200 MB hard drive space
- Internet connection
- USB 1.1 port (2.0 recommended)

Ultrasonic sensors:

The ultrasonic sensors are used for the obstacle avoidance feature. The ultrasonic sensor that is used in the project has the following configuration.

- High sensitivity ultrasonic range sensor

- Working Voltage - DC 5V, Working Current - 15mA
- Max Range - 4m, Min Range - 2cm, Measuring Angle - 15 degree

Mini keyboard:

The mini keyboard would act as a hand held remote through which the user can select a particular feature to execute and hear the result. The keyboard used in this project has the following configuration:

- 2.4GHz Mini Wireless QWERTY keyboard, Touchpad combo, multimedia control keys and PC gaming control keys with USB interface adapter.
- 92 keys Wireless QWERTY Keyboard, Touchpad which supports multi-finger functions, A single finger click as left mouse function, two-finger click as the right mouse function, double finger drags as the rolling screen.
- Built-in highly sensitive smart touchpad with 360-degree flip design. Innovative shape, portable, elegant. The Ergonomically handheld design is easy to carry and operate.
- Perfect for PC, Pad, Android TV Box, Google TV Box, Xbox 360, PS3, HTPC, IPTV.

3.2.2 Software Requirements

The software requirements deal with the prerequisites and software resource requirements that are needed to be installed on the computer to give optimal functioning of the application.

- Operating System – Raspbian Stretch
- Programming Language – Python2, Python3, Shell scripting
- IDE – Geany IDE
- Libraries – Pillow, OpenCV, datetime, Geocoder, numpy, scipy, tesseract, gTTS, IMAP protocol, playsound, News API, Open Weather API, email library.

3.3 Input Requirements

The features like voice out current date and time, voice out current location, voice out the current weather, voice out the top 10 news headlines just requires the push of corresponding button from the hand held remote.

The voice out the unread email requires the user to have a configured email address which would allow the IMAP call from the python code. The OCR feature and the dominant color feature requires an image to be taken by the user using the webcam hosted on the product. The obstacle detection feature takes the input feed from the ultrasonic sensor and sets an alarm to high if a particular object is too close to the user. The product must have a good internet connection for the various API's work efficiently.

3.4 Output Requirements

The product must have a good internet connection for the various API's work efficiently. The product must have an audio output connected to it which can be in the form of headphones or Bluetooth audio outputs. The voice output results for all the features are played to the user or the visually impaired person through the audio output devices. The volume needs to be high in order for the user to hear the results given from the device.

3.5 Language Specification

The language specification is used during system analysis, system design and requirement analysis to describe the system at a higher level than the programming language, and hence used to the executable code for the system. These area unit are usually in a roundabout way dead. they're meant to explain the what, not the however. Indeed, it's thought-about as a slip-up if a demand specification is untidy with inessential implementation detail.

A common elementary assumption of the many specification approaches is that programs area unit modelled as pure mathematics or model-theoretic structures that embrace a set of sets of knowledge values at the side of functions over those sets. This level of abstraction coincides with the read that the correctness of the input/output behavior of a program takes precedence over all its alternative properties.

3.5.1 Python

Python is a multi paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution. Python's design offers some support for functional programming in the Lisp tradition. It has filter, map, and reduce functions; list comprehensions, dictionaries, sets, and generator expressions. The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach.

Python strives for a simpler, less-cluttered syntax and grammar while giving developers a choice in their coding methodology. In contrast to Perl's "there is more than one way to do it" motto,

Python embraces a "there should be one—and preferably only one—obvious way to do it" design philosophy. Alex Martelli, a Fellow at the Python Software Foundation and Python book author, writes that "To describe something as 'clever' is not considered a compliment in the Python culture."

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of the CPython reference implementation that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. Cython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter.

An important goal of Python's developers is keeping it fun to use. This is reflected in the language's name—a tribute to the British comedy group Monty Python—and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (from a famous Monty Python sketch) instead of the standard foo and bar.

A common neologism in the Python community is pythonic, which can have a wide range of meanings related to program style. To say that code is pythonic is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called unpythonic. Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as Pythonistas.

3.5.2 Shell Scripting

A shell script is a computer program designed to be run by the Unix shell, a command-line interpreter.The various dialects of shell scripts are considered to be scripting languages. Typical operations performed by shell scripts include file manipulation, program execution, and printing

text. A script which sets up the environment, runs the program, and does any necessary cleanup, logging, etc. is called a wrapper. The term is also used more generally to mean the automated mode of running an operating system shell; in specific operating systems they are called other things such as batch files (MSDos-Win95 stream, OS/2), command procedures (VMS), and shell scripts (Windows NT stream and third-party derivatives like 4NT—article is at cmd.exe), and mainframe operating systems are associated with a number of terms.

The typical Unix/Linux/POSIX-compliant installation includes the KornShell (ksh) in several possible versions such as ksh88, Korn Shell '93 and others. The oldest shell still in common use is the Bourne shell (sh); Unix systems invariably also include the C shell (csh), Bash (bash), a Remote Shell (rsh), a Secure Shell (ssh) for SSL telnet connections, and a shell which is a main component of the Tcl/Tk installation usually called tclsh; wish is a GUI-based Tcl/Tk shell. The C and Tcl shells have syntax quite similar to that of said programming languages, and the Korn shells and Bash are developments of the Bourne shell, which is based on the ALGOL language with elements of a number of others added as well. On the other hand, the various shells plus tools like awk, sed, grep, and BASIC, Lisp, C and so forth contributed to the Perl programming language.

Other shells available on a machine or available for download and/or purchase include Almquist shell (ash), PowerShell (msh), Z shell (zsh, a particularly common enhanced KornShell), the Tenex C Shell (tcsh), a Perl-like shell (psh). Related programs such as shells based on Python, Ruby, C, Java, Perl, Pascal, Rexx &c in various forms are also widely available. Another somewhat common shell is osh, whose manual page states it "is an enhanced, backward-compatible port of the standard command interpreter from Sixth Edition UNIX." Windows-Unix interoperability software such as the MKS Toolkit, Cygwin, UWIN, Interix and others make the above shells and Unix programming available on Windows systems, providing functionality all the way down to signals and other inter-process communication, system calls and APIs. Microsoft distributes Windows Services for UNIX for use with its NT-based operating systems in particular, which have a POSIX environmental subsystem.

Chapter 4

SYSTEM DESIGN AND ANALYSIS

4.1 Preliminary Design

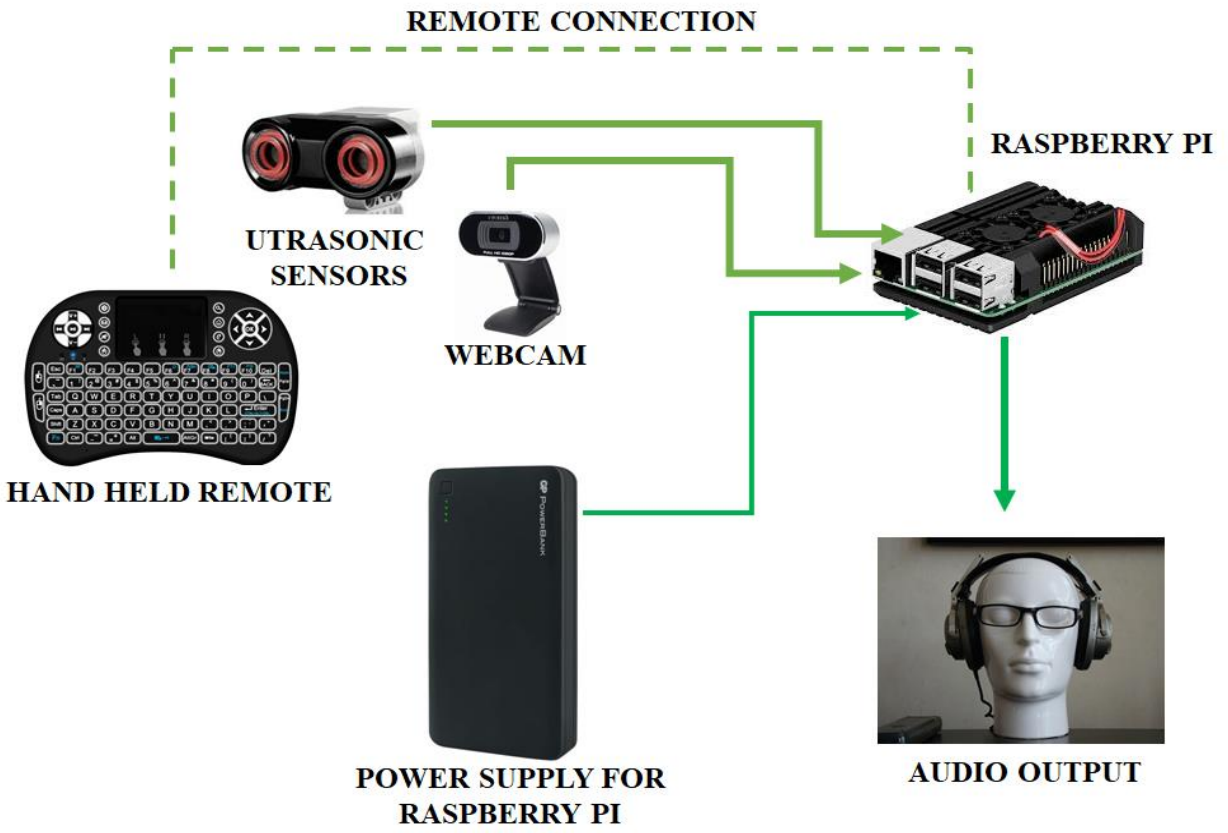


Fig. Design outline of the product

The hand held remote is the remote that is held by the visually impaired user for providing the input for a certain feature to execute by the raspberry pi processor. The webcam and the ultrasonic sensors are mounted over the glasses which would be worn by the visually impaired

user. The ultrasonic sensors are used to analyze the environment for the obstacles and alert the user when an object is too close to the user. The webcam is used too take images for the OCR and color detection features that are provided by the product. The power bank is used to power the raspberry pi processor. The raspberry pi processor is the central processing using where all the features processing occurs. The results of all the features are provided to the visually impaired person using the audio output which can be in the form of a headphones.

4.2 System Architecture

1. Voice out the current date and time

The smart vision system uses the datetime module to get the current date and time and it is stored into a text document. The text document that is generated is later given as an input to the gTTS (google text to speech module) which converts the text to speech and the speech output is stored into a .mp3 file and the .mp3 file is played out as the output to the visually impaired person using the speakers.

2. Voice out the current location

The smart vision system uses the ip address for knowing the current location, region and the country. The shell command curl ipinfo is used to get the current location, region and the country. The result is stored into a text document and the text document is given as an input to the gTTS which converts the text into .mp3 file. The .mp3 file is played back to the visually impaired person.

3. Voice out the current temperature

The smart vision system uses the OpenWeatherMap API to get the current temperature and the weather description for the day. The result is stored into a text document and this document is given to gTTS which converts the result into speech format. The speech result is then played to the visually impaired person.

4. Voice out the new emails

The smart vision system has the capability to read out the new emails arrived to the person's email. The vision system uses the the imap module to connect the SMTP server over the SSL. Using already feed in user credentials we login to the mail server and get the unread messages and store it into a text file. Using gTTS the text file is converted into a speech format and the .mp3 file is played out to the visually impaired person.

5. Voice out the top 10 headlines for the day

The smart vision system uses the News API to get the top 10 news headlines for the day. The result is stored into a text document and the text document is given as an input to the gTTS. The gTTS converts it into the .mp3 file which is played to the visually impaired person.

6. Voice the text content present in a picture

The webcam is used to take a picture of the document or any text containing object. The picture is taken and tesseract ocr module is used to extract the text from the image and the extracted text is stored into a text document. The gTTS is used to convert the text to speech format. The speech is then played back to the visually impaired person.

7. Voice out the dominant color in a picture.

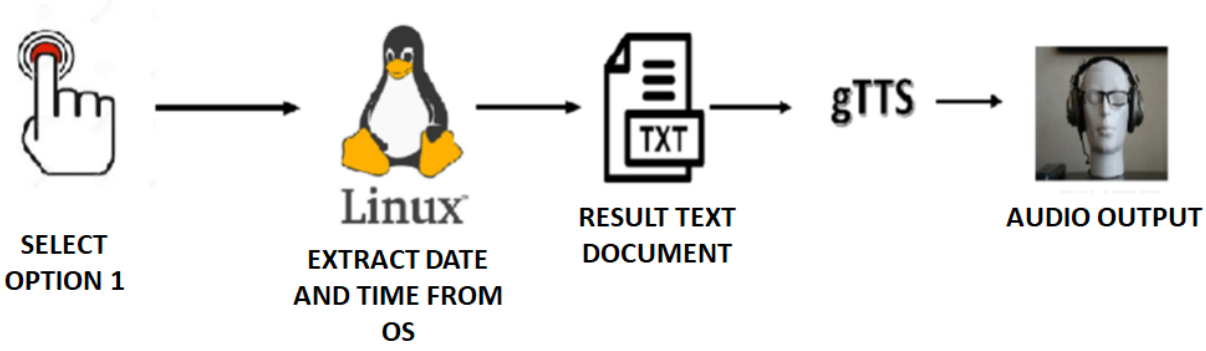
For the dominant color feature an image is captured form the webcam and this image is given to the python code which detects the dominant color in the image. The result is stored into a text document and this text document is given as an input to the gTTS program which converts the text results into a .mp3 file which is played to the visually impaired person via the audio output device which here is headphones.

8. Obstacle detection

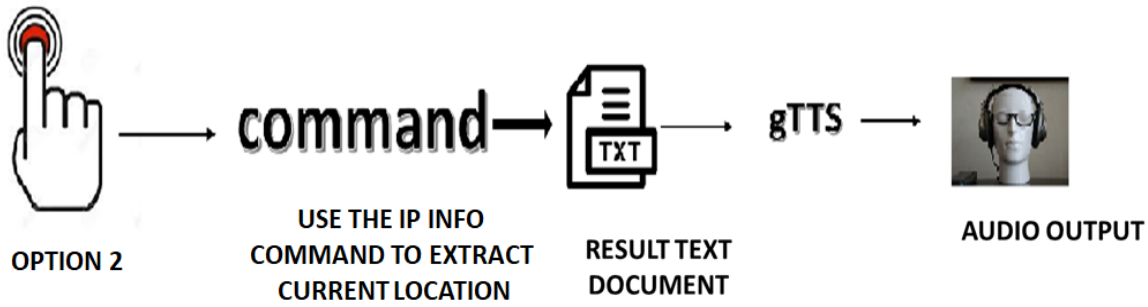
The ultrasonic sensors are used to analyze the environment for the obstacles and alert the user when an object is too close to the user.

SYSTEM ARCHITECTURE FOR EACH FEATURE

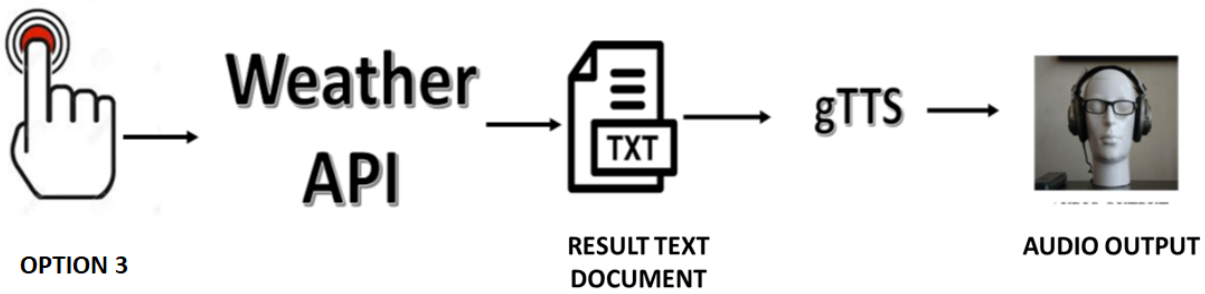
1. Option 1: Voice out the current date and time



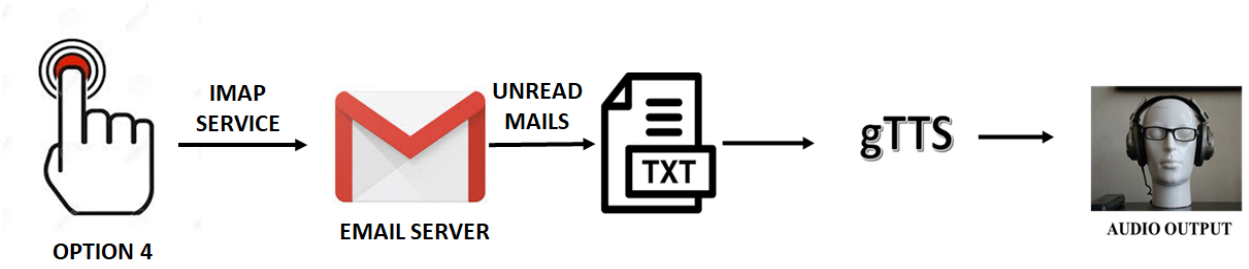
2. Option 2: Voice out the current location



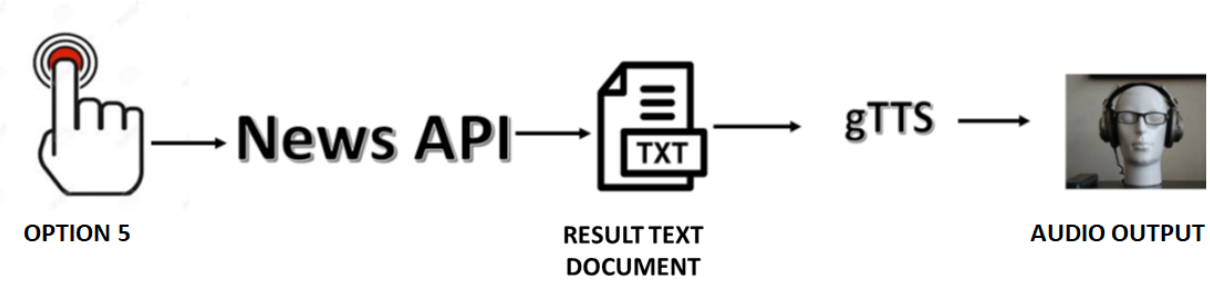
3. Option 3: Voice out the current weather



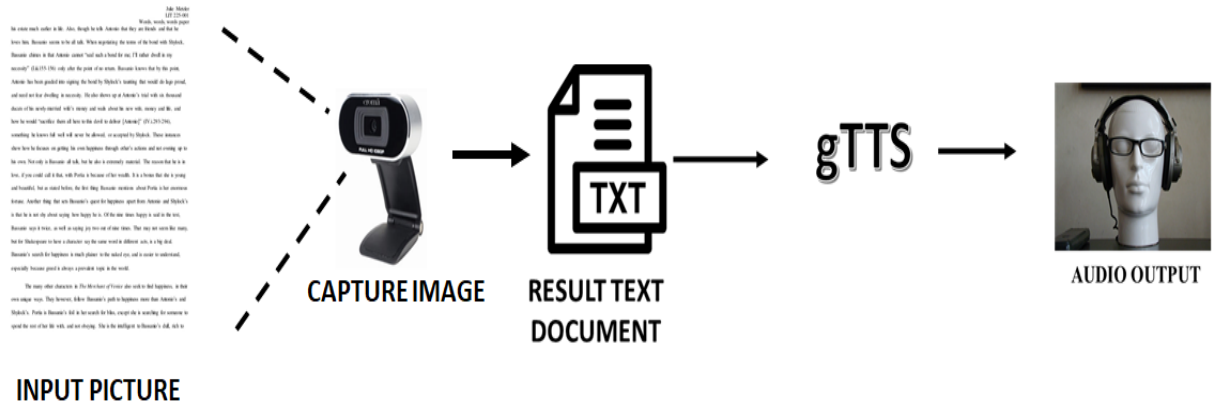
4. Option 4: Voice out the unread emails



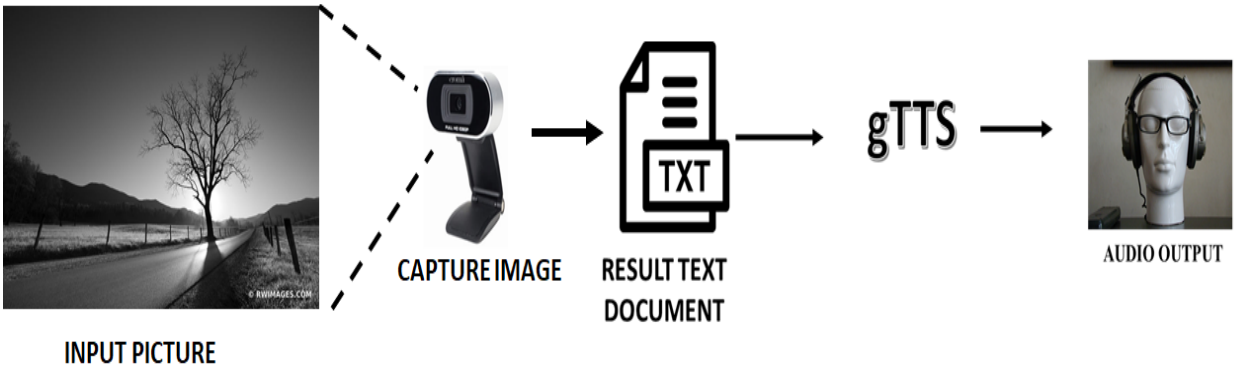
5. Option 5: Voice out the top 10 news headlines



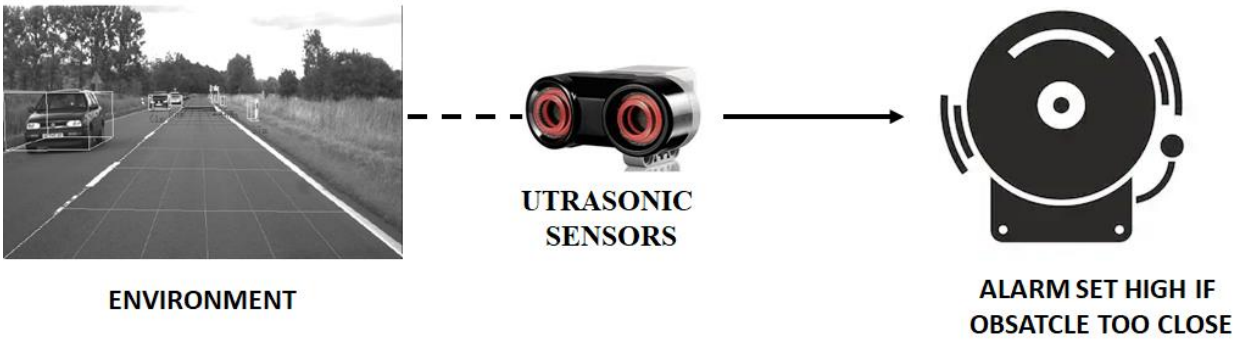
6. Option 6: Voice out the text in an image



7. Option 7 : Voice out the dominant color in an image



8. Obstacle detection



4.3 Data Flow Diagram

4.3.1 System DFD

This diagram represents the entire systems flow of data and how each module communicates with the other. The data flow diagram for the smart glasses for the visually impaired people starts by taking the input option from the remote keypad and then the particular feature is processed by the raspberry pi processor. The data flow diagram for the developed project is as shown below:

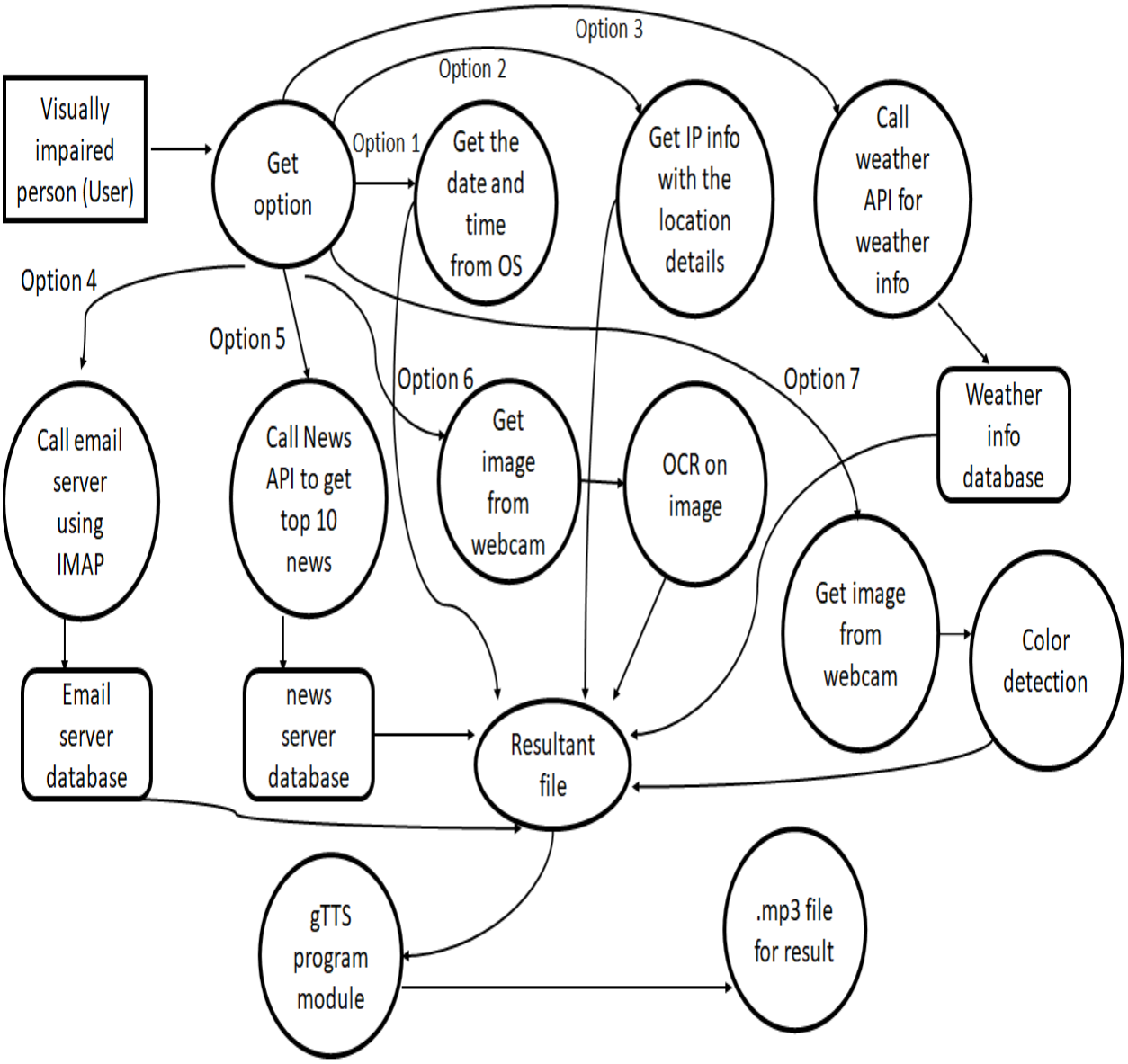


Fig: DFD design 1

The above diagram shows the data flow diagram for the features voice out the current data and time, voice out the current weather information, voice out the current location, voice out the unread emails, voice out the top 10 news headlines, voice out the text in an image and voice out the dominant color In the image.

The data flow diagram for the obstacle avoidance is as shown below:

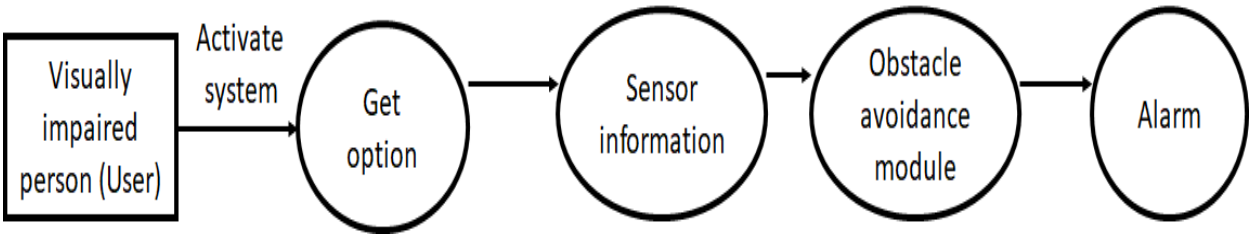
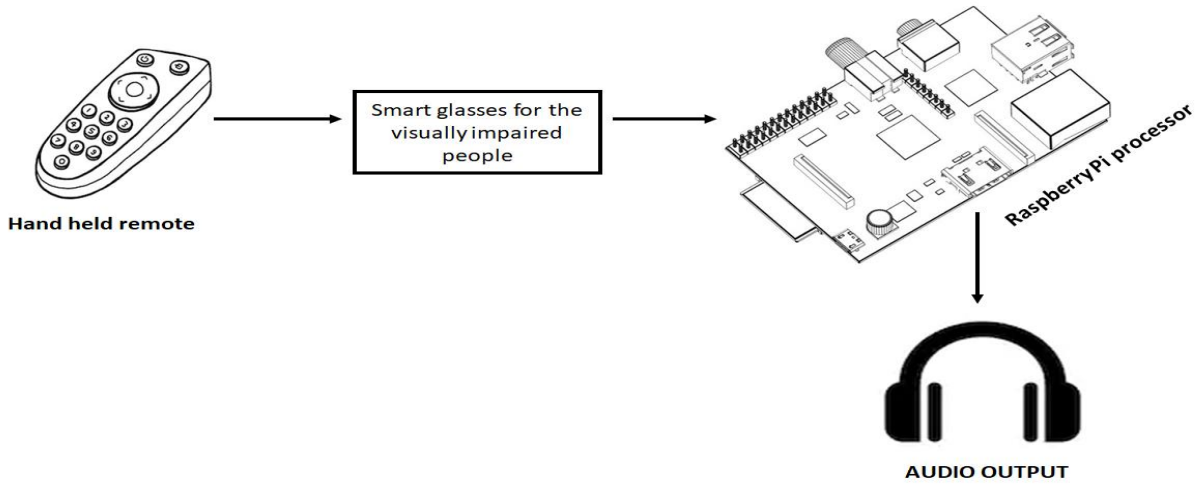


Fig: DFD design 2

4.3.2 Input DFD

The input DFD depicts the ways the app accepts input from the user to carry out the execution of various features in the product. The only input that is required from the visually impaired person or the user of the product is the press of the button in the hand held remote which corresponds to a particular feature. Each and every button on the hand held remote corresponds to different feature. When a particular button is pressed the corresponding feature is executed by the raspberry pi and the output or the result of the execution is given out in the form an audio output which is heard by the visually impaired person through the audio output device.



4.4 Use Case Diagram

In the following diagram, various use cases of “Smart glasses for the visually impaired people” is depicted. The first use case of the system is voicing out the current date and the time for the user. The second use case is voicing out the current location to the user. The third use case of the system is voicing out the current weather information for the user. The fifth use case of the system is voicing out the unread emails to the user. The sixth use case is to voice out the top ten news headlines for the day. The seventh use case of the system is to voice out the text in a image using OCR. The eighth use case of the system is to voice out the dominant color in an image. The ninth use case of the system is providing the obstacle avoidance system using the sensors.

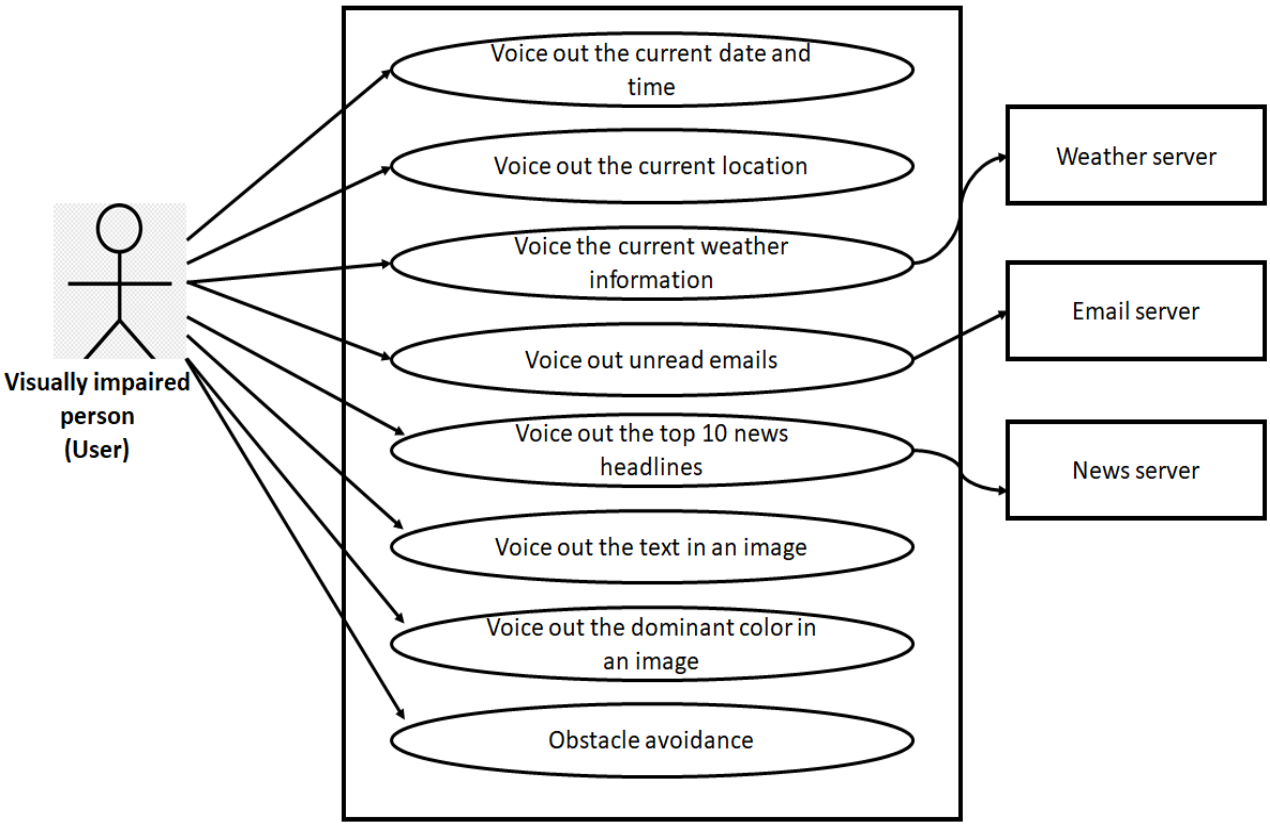


Fig: Use case diagram of the system

Chapter 5

Implementation

5.1 Different Modules of the project

The various modules present in the project along with the code implemented for the particular feature are given below:

1. Date and time module

```
import time

import os

import pytesseract

import pyttsx3

from PIL import Image

from googletrans import Translator

from gtts import gTTS

from playsound import playsound

from datetime import datetime

from datetime import date

from datetime import date


f = open("/home/pi/Desktop/final_year_project/date_and_time/date_time_text.txt",'w')

today = date.today()
```

```
d2 = today.strftime("%B %d, %Y")

print("d2 =", d2)

f.write("Today's date is : ")

f.write(d2)

f.write("\n")

t = time.localtime()

current_time = time.strftime("%H:%M:%S",t)

print(current_time)

f.write("Current time is : ")

f.write(current_time)

f.write("\n")

print("Have a nice day")

f.write("Have a nice day")

f.close()
```

2. Current location module

```
#!/usr/bash
rm /home/pi/Desktop/final_year_project/current_location/location.txt

printf "current city is : " >>
/home/pi/Desktop/final_year_project/current_location/location.txt
curl ipinfo.io/city >>
/home/pi/Desktop/final_year_project/current_location/location.txt
printf "\n"
```

```
printf "current region is : " >>
/home/pi/Desktop/final_year_project/current_location/location.txt
curl ipinfo.io/region >>
/home/pi/Desktop/final_year_project/current_location/location.txt
printf "\n"

printf "Current country is : " >>
/home/pi/Desktop/final_year_project/current_location/location.txt
curl ipinfo.io/country >>
/home/pi/Desktop/final_year_project/current_location/location.txt
printf "\n"

echo "Program complete ,done" >>
/home/pi/Desktop/final_year_project/current_location/location.txt
```

3. Current weather information

```
import requests, json

api_key = "f28e654cf78a932a10e58b89e87bb4aa"

base_url = "http://api.openweathermap.org/data/2.5/weather?"

city_name = "Bangalore"


complete_url = base_url + "appid=" + api_key + "&q=" + city_name

response = requests.get(complete_url)

x = response.json()


if x["cod"] != "404":
```

```
      y = x["main"]

      current_temperature = y["temp"]

      current_pressure = y["pressure"]

      current_humidiy = y["humidity"]


      z = x["weather"]

      weather_description = z[0]["description"]

      # print following values

      print(" Temperature (in kelvin unit) = " +

                              str(current_temperature) +

            "\n atmospheric pressure (in hPa unit) = " +

                              str(current_pressure) +

            "\n humidity (in percentage) = " +

                              str(current_humidiy) +

            "\n description = " +

                              str(weather_description))

      f =
open("/home/pi/Desktop/final_year_project/weather/weather_text.txt","w")

      f.write("Today's temperature : "+str(current_temperature))

      f.write("\nHumidity : "+str(current_humidiy))

      f.write("\nDescription : "+str(weather_description))

      f.write("\nProgram complete ,done")

      f.close()

else:
```

```
      print(" City Not Found ")
```

4. Unread emails module

```
import email
import imaplib
import os

mail = imaplib.IMAP4_SSL('imap.gmail.com')
(retcode, capabilities) =
mail.login('wingsoffirew@gmail.com','wingswings')
mail.list()
mail.select('inbox')

n=0
(retcode, messages) = mail.search(None, '(UNSEEN)')
if retcode == 'OK':

   for num in messages[0].split() :

os.remove('/home/pi/Desktop/final_year_project/email/emailemail_text.txt')
      print ('Processing ')
      n=n+1
      typ, data = mail.fetch(num,'(RFC822)')
      for response_part in data:
         if isinstance(response_part, tuple):
             original = email.message_from_bytes(response_part[1])

            # print (original['From'])
            # print (original['Subject'])
             raw_email = data[0][1]
             raw_email_string = raw_email.decode('utf-8')
```

```
            email_message = email.message_from_string(raw_email_string)
            for part in email_message.walk():
                        if (part.get_content_type() == "text/plain"): #
ignore attachments/html
                              body = part.get_payload(decode=True)
                              #save_string =
str(r"/home/pi/Desktop/final_year_project/email" + str('email_text') +
".txt" )

save_string='/home/pi/Desktop/final_year_project/email/emailemail_text.txt
'
                              myfile = open(save_string, 'a')
                              myfile.write("email from : ")
                              myfile.write(original['From']+'\n')
                              myfile.write("email subject: ")
                              myfile.write(original['Subject']+'\n')
                              myfile.write("email is :\n ")
                              myfile.write(body.decode('utf-8')+"\n
Program complete, done")
                              myfile.close()
                        else:
                              continue

            typ, data = mail.store(num,'+FLAGS','\\Seen')
print (n)
if(n==0):
       #os.remove('/home/pi/Desktop/final_year_project/email/emailemail_te
xt.txt')
       save_string='/home/pi/Desktop/final_year_project/email/emailemail_t
ext.txt'
       myfile = open(save_string, 'w')
       myfile.write("No new emails arrived,done")
       myfile.close()
```

```
        print("No new emails arrived")
```

5. Top 10 news headlines module

```
import requests
import os
from gtts import gTTS
import time



def NewsFromBBC():

    # BBC news api
    main_url = " https://newsapi.org/v1/articles?source=bbc-
news&sortBy=top&apiKey=dc8e4f0486b84d01bf7c18d5d8f06e72"

    # fetching data in json format
    open_bbc_page = requests.get(main_url).json()

    # getting all articles in a string article
    article = open_bbc_page["articles"]

    # empty list which will
    # contain all trending news
    results = []

    for ar in article:
        results.append(ar["title"])

    f=
open("/home/pi/Desktop/final_year_project/news_headlines/news_text.txt","w
+")
    for i in range(len(results)):
```

```
        # printing all trending news
        s = "headline "+str(i + 1)+": "+ results[i]
        print(s)
        f.write(s+"\n")
        time.sleep(1)
    f.write("Program complete, complete \n")
    f.close()


# Driver Code
if __name__ == '__main__':

    # function call
    NewsFromBBC()
```

6. OCR module

```
import cv2
import pytesseract
import os

try:
    f =
open("//home//pi//Desktop//final_year_project//ocr//saved_img.jpg")
    f2 = open("//home//pi//Desktop//final_year_project//ocr//saved_img-
final.jpg")
    f3 =
open("//home//pi//Desktop//final_year_project//ocr//ocr_text.txt")
    # Do something with the file
    print("file exists")

os.remove("//home//pi//Desktop//final_year_project//ocr//saved_img.jpg")
```

```
    os.remove("//home//pi//Desktop//final_year_project//ocr//saved_img-
final.jpg")

os.remove("//home//pi//Desktop//final_year_project//ocr//ocr_text.txt")
    #os.remove("ocr_voice.mp3")

except IOError:
    print("Files no found")
finally:
    print("complete")

key = cv2. waitKey(1)
webcam = cv2.VideoCapture(0)
while True:
    try:
        check, frame = webcam.read()
        print(check) #prints true as long as the webcam is running
        print(frame) #prints matrix values of each framecd
        cv2.imshow("Capturing", frame)
        key = cv2.waitKey(1)
        if key == ord('s'):

cv2.imwrite(filename='//home//pi//Desktop//final_year_project//ocr//saved_
img.jpg', img=frame)
            webcam.release()
            img_new =
cv2.imread('//home//pi//Desktop//final_year_project//ocr//saved_img.jpg',
cv2.IMREAD_GRAYSCALE)
            img_new = cv2.imshow("Captured Image", img_new)
            cv2.waitKey(1650)
            cv2.destroyAllWindows()
            print("Processing image...")
```

```
            img_ =
cv2.imread('//home//pi//Desktop//final_year_project//ocr//saved_img.jpg',
cv2.IMREAD_ANYCOLOR)
            print("Converting RGB image to grayscale...")
            gray = cv2.cvtColor(img_, cv2.COLOR_BGR2GRAY)
            print("Converted RGB image to grayscale...")
            print("Resizing image to 28x28 scale...")
            img_ = cv2.resize(gray,(999,999))
            print("Resized...")
            img_resized =
cv2.imwrite(filename='//home//pi//Desktop//final_year_project//ocr//saved_
img-final.jpg', img=img_)
            print("Image saved!")
            print("Hello we are printing here")
            text =
pytesseract.image_to_string('//home//pi//Desktop//final_year_project//ocr/
/saved_img.jpg').encode('utf-8').strip()
            print(text)
            text_file =
open("//home//pi//Desktop//final_year_project//ocr//ocr_text.txt","w")
            n = text_file.write(text)
            text_file.write("Program complete, complete")
            text_file.close()
            #execfile('he4.py')

            break
        elif key == ord('q'):
            print("Turning off camera.")
            webcam.release()
            print("Camera off.")
            print("Program ended.")
            cv2.destroyAllWindows()
            break
```

```
    except(KeyboardInterrupt):
        print("Turning off camera.")
        webcam.release()
        print("Camera off.")
        print("Program ended.")
        cv2.destroyAllWindows()
        break
```

7. Dominant color feature

```
from colorthief import ColorThief
import numpy as np
from scipy import spatial

color_thief =
ColorThief('/home/pi/Desktop/final_year_project/color/img.jpg')
# get the dominant color
dominant_color = color_thief.get_color(quality=1)
palette = color_thief.get_palette(color_count=6)
a = dominant_color[0]
b = dominant_color[1]
c = dominant_color[2]

print(a)
print(b)
print(c)
#!/usr/bin/env python
# rgb2colorname.py
# by wilsonmar@gmail.com, ayush.original@gmail.com,
https://github.com/paarthneekhara
# Usage: python rgb2colorname.py
# To ensure this program has no external dependencies,
```

```
# an array and dictionary is used in place of I/O from input reference
files.
# Explained in
https://github.com/jetbloom/rgb2colorname/blob/master/README.md

# import Algorithmia



# TODO: define function for use in Algorithmia.com or other API:
#def find_nearest_vector(array, value):
#  #
http://docs.scipy.org/doc/numpy/reference/generated/numpy.linalg.norm.html
#  idx = np.array([np.linalg.norm(x+y) for (x,y,z) in array-
value]).argmin()
#  return array[idx]

# NO import importlib & import_module('rgbcsv2rgbarray.py.txt') to avoid
external dependencies.
#moduleName='rgbcsv2rgbarray.py.txt'
# mport_module(moduleName)
#### Paste in contents of rgb_combined_v01.csv.txt below: ###
# 2016-08-31-07:53 (local time) rgbcsv2rgbarray.py START: outrowcount=570.
RGB = np.array([ \
 [0,0,0] \
,[0,0,128] \
,[0,0,139] \
,[0,0,139] \
,[0,0,205] \
,[0,0,205] \
,[0,0,238] \
,[0,0,255] \
,[0,0,255] \
,[0,100,0] \
```

```
,[0,104,139] \
,[0,128,0] \
,[0,128,0] \
,[0,128,128] \
,[0,134,139] \
,[0,139,0] \
,[0,139,69] \
,[0,139,139] \
### End of paste ###

# Dictionary of colornames indexed by key _Hex:
# See https://bdhacker.wordpress.com/2010/02/27/python-tutorial-
dictionaries-key-value-pair-maps-basics/
HexNameDict = { \
,"#006400":"DarkGreen" \
,"#00688B":"DeepSkyBlue4" \
,"#008000":"WebGreen" \
,"#008000":"green" \
,"#008080":"teal" \
,"#00868B":"turquoise4" \
,"#008B00":"green4" \
,"#FFFFFF":"white" \
,"#FFFFFF":"gray100" \
}

# TODO: Test calls using variety of RGB input values
# TODO: Change to call argument with the point to find
# pt = [221,183,134] # approximate to
#pt = [0,0,0] # example needing zerofill
pt = [a,b,c] # = "burlywood","#DEB887"]
# pt = [222,184,135] # = "burlywood","#DEB887"]
# pt = [154,205,50] # = OliveDrab
```

```
# Lookup color name using Hex:ColorName dictionary:
NearestRGB = (RGB[spatial.KDTree(RGB).query(pt)[1]])

# TODO: Calculate Hex from pt. (upper case letters)
# Instead of str(hex(pt[0])[2:]) in Python2, this is Python3 compatible:
s = '#' \
    + format(NearestRGB[0],'x').zfill(2) \
    + format(NearestRGB[1],'x').zfill(2) \
    + format(NearestRGB[2],'x').zfill(2)
ColorHex = s.upper() # "#8B7355"  # "#8B7355"
ColorDiff = \
     '('+'{0:+d}'.format(NearestRGB[0]-pt[0]) \
    +','+'{0:+d}'.format(NearestRGB[1]-pt[1]) \
    +','+'{0:+d}'.format(NearestRGB[2]-pt[2]) \
    +')'
try: ## TODO: try catch block per
https://wiki.python.org/moin/HandlingExceptions
    ColorName=HexNameDict[ColorHex]
except:
       ColorName="not found"
print ('Nearest color name to input RGB ' \
    + str(pt) \
    + ' is "'+ ColorName +'"' \
    +' '+ ColorHex  \
    +' '+ str(NearestRGB) \
    +', '+ ColorDiff \
    +'.')
f = open('/home/pi/Desktop/final_year_project/color/color_text.txt',"w")
f.write("Dominant color in the image is : \n")
f.write(ColorName)
f.close()
```

8. Shell script – Controller program

```
#!/usr/bash
play
/home/pi/Desktop/final_year_project/extravoices/welcome_speech.m
p3
while $1
do
sleep 1
echo "Please enter your option"
play
/home/pi/Desktop/final_year_project/extravoices/get_option.mp3
read option
case ${option} in
   1) echo "option 1 choosen : funtion date and time"
      play
/home/pi/Desktop/final_year_project/extravoices/option1.mp3
      python3
/home/pi/Desktop/final_year_project/date_and_time/date_time.py
&&
      python3
/home/pi/Desktop/final_year_project/date_and_time/date_time_gtts
.py &&
      play
/home/pi/Desktop/final_year_project/date_and_time/date_time_voic
e.mp3 &&
      play
/home/pi/Desktop/final_year_project/extravoices/program_complete
.mp3
      sleep 1.5
```

```
      ;;
   2) echo "option 2 choosen : function current location"
      play
/home/pi/Desktop/final_year_project/extravoices/option2.mp3
      sh
/home/pi/Desktop/final_year_project/current_location/current_loc
.sh &&
      python3
/home/pi/Desktop/final_year_project/current_location/location_gt
ts.py &&
      play
/home/pi/Desktop/final_year_project/current_location/cur_loc.mp3
&&
      play
/home/pi/Desktop/final_year_project/extravoices/program_complete
.mp3
      sleep 1.5
      ;;

   7) echo "option 7 choosen : function current weather"
      play
/home/pi/Desktop/final_year_project/extravoices/temperature.mp3
      python3
/home/pi/Desktop/final_year_project/weather/weather_code.py &&
      python3
/home/pi/Desktop/final_year_project/weather/weather_gtts.py &&
      play
/home/pi/Desktop/final_year_project/weather/weather_voice.mp3 &&
```

```
      play
/home/pi/Desktop/final_year_project/extravoices/program_complete
.mp3
      sleep 1.5
      ;;

   3) echo "option 3 choosen : function email reading"
      play
/home/pi/Desktop/final_year_project/extravoices/option3.mp3
      python3
/home/pi/Desktop/final_year_project/email/email_code.py &&
      python3
/home/pi/Desktop/final_year_project/email/email_gtts.py &&
      play
/home/pi/Desktop/final_year_project/email/email_voice.mp3 &&
      play
/home/pi/Desktop/final_year_project/extravoices/program_complete
.mp3
      sleep 1.5
      ;;

   4) echo "option 4 choosen : newsheadlines"
      play
/home/pi/Desktop/final_year_project/extravoices/option4.mp3
      python3
/home/pi/Desktop/final_year_project/news_headlines/news.py &&
      python3
/home/pi/Desktop/final_year_project/news_headlines/news_gtts.py
&&
```

```
      play
/home/pi/Desktop/final_year_project/news_headlines/news_voice.mp
3 &&
      play
/home/pi/Desktop/final_year_project/extravoices/program_complete
.mp3
      sleep 1.5
      ;;

   5) echo "option 5 choosen : ocr"
      play
/home/pi/Desktop/final_year_project/extravoices/option5.mp3
      python
/home/pi/Desktop/final_year_project/ocr/ocr_img_to_text.py &&
      python3
/home/pi/Desktop/final_year_project/ocr/ocr_gtts.py &&
      play /home/pi/Desktop/final_year_project/ocr/ocr_voice.mp3
&&
      play
/home/pi/Desktop/final_year_project/extravoices/program_complete
.mp3
      sleep 1.5
      ;;

   6) play
/home/pi/Desktop/final_year_project/extravoices/help_welcome.mp3
      play
/home/pi/Desktop/final_year_project/extravoices/help_info.mp3
```

```
      play
/home/pi/Desktop/final_year_project/extravoices/help_again.mp3
       sleep 1.5
       ;;


   8) play
/home/pi/Desktop/final_year_project/extravoices/color_welcome.mp
3
      python
/home/pi/Desktop/final_year_project/color/capture_image.py
      play
/home/pi/Desktop/final_year_project/extravoices/color2.mp3
      python3
/home/pi/Desktop/final_year_project/color/color_detect.py
      python3
/home/pi/Desktop/final_year_project/color/color_gtts.py
      play
/home/pi/Desktop/final_year_project/color/color_voice.mp3
       sleep 1.5
       ;;

   9) echo "Program complete"
       play
/home/pi/Desktop/final_year_project/extravoices/exit.mp3
       exit 1
       ;;

   *)  echo "improper button please choosen again"
```

```
      play
/home/pi/Desktop/final_year_project/extravoices/improper_button.
mp3
      sleep 1.5
      ;;
esac
done
```

9. gTTS module

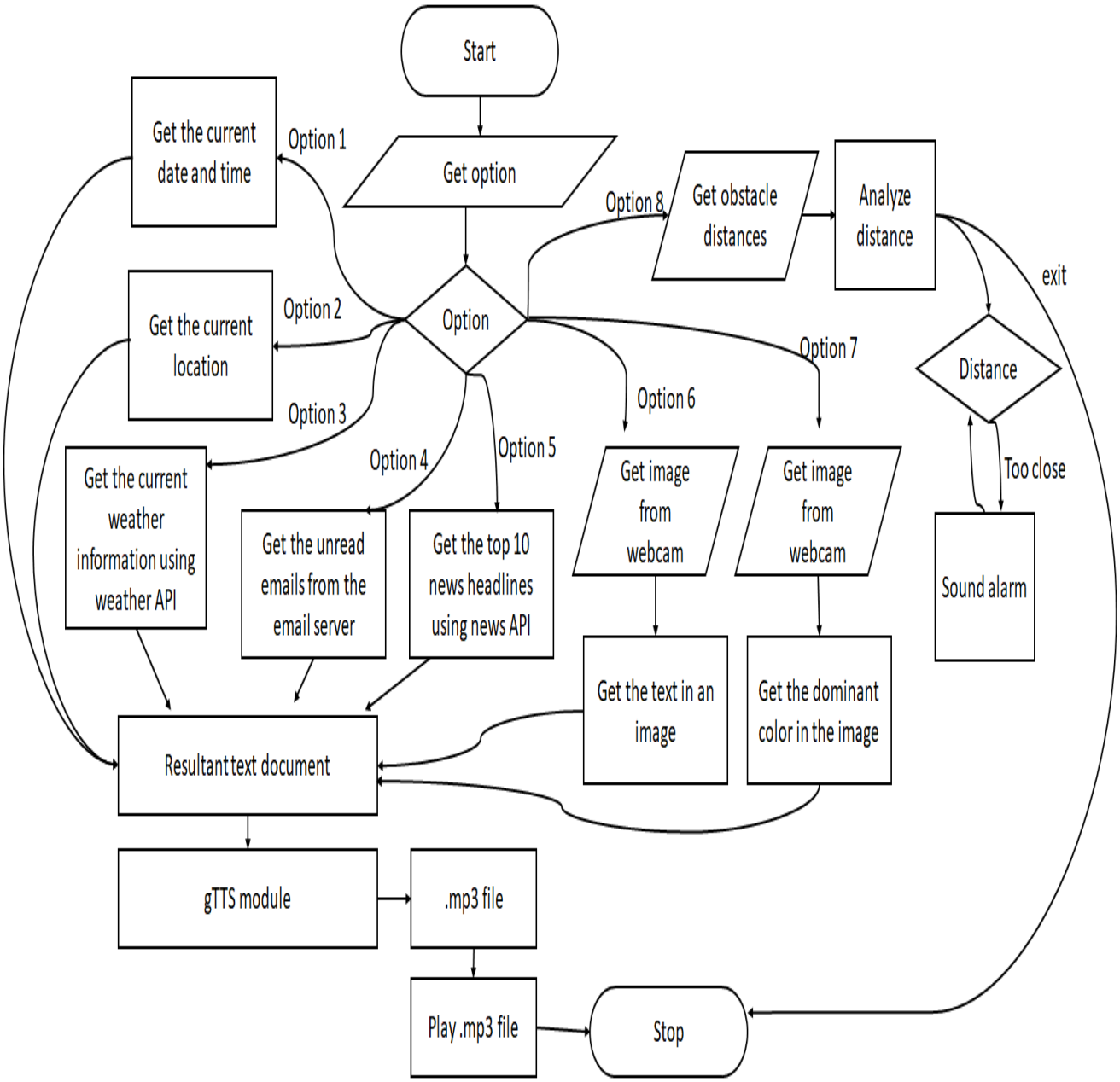
```
from gtts import gTTS
import os
file =
open("//home//pi//Desktop//final_year_project//color//color_text.txt","r")
.read().replace("\n"," ")
speech = gTTS(text = str(file),lang = 'en',slow = False)
speech.save("//home//pi//Desktop//final_year_project//color//color_voice.m
p3")
```

5.2 Dependencies

The various dependencies for the opencv library used for implementing system is shown below:

- sudo apt install build-essential cmake git pkg-config libgtk-3-dev "libcanberra-gtk*"
- sudo apt install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev libxvidcore-dev libx264-dev
- sudo apt install libjpeg-dev libpng-dev libtiff-dev gfortran openexr libatlas-base-dev opencl-headers
- sudo apt install python3-dev python3-numpy libtbb2 libtbb-dev libdc1394-22-d

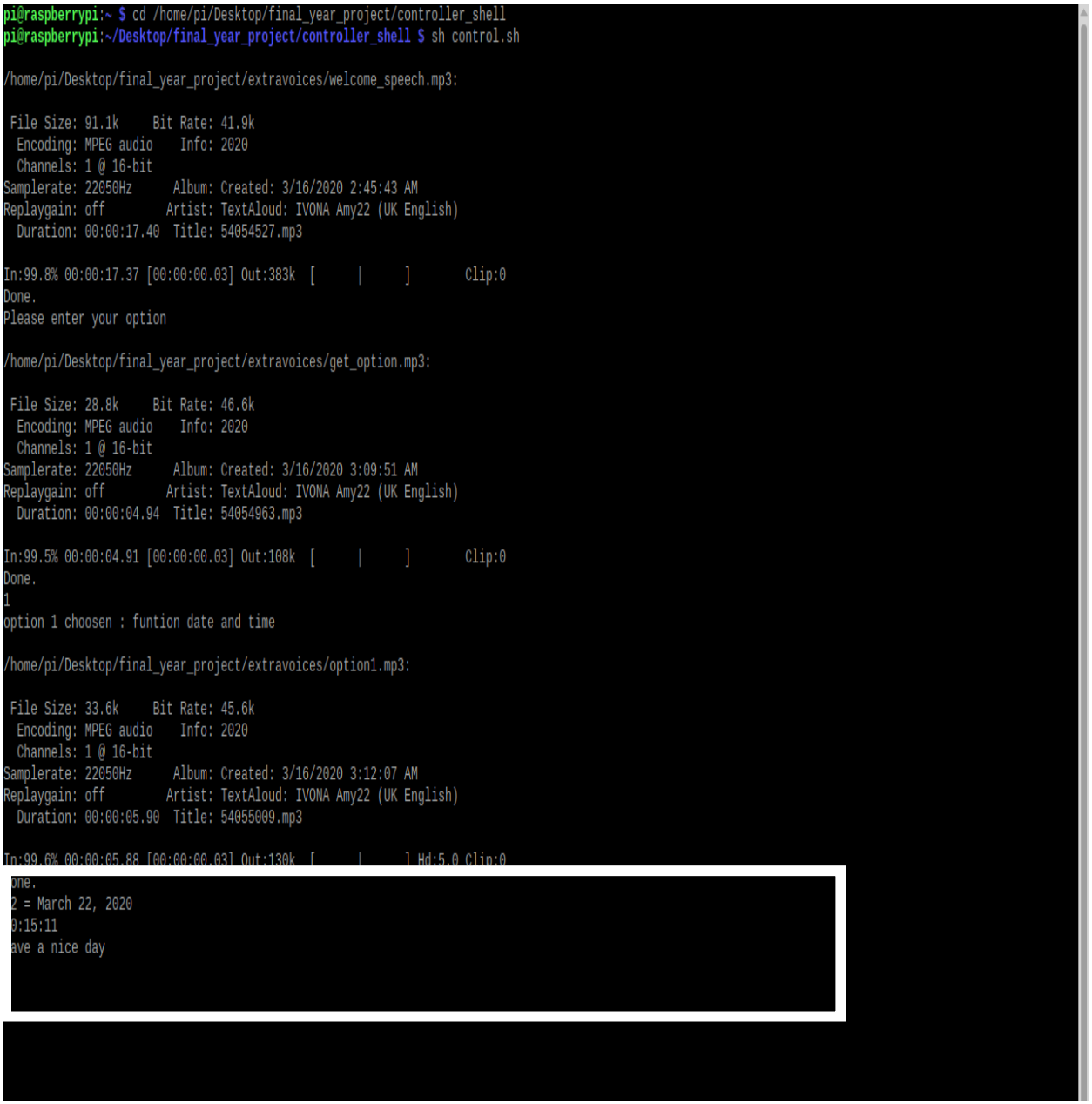
5.3 Flow Chart of Proposed system



Chapter 6

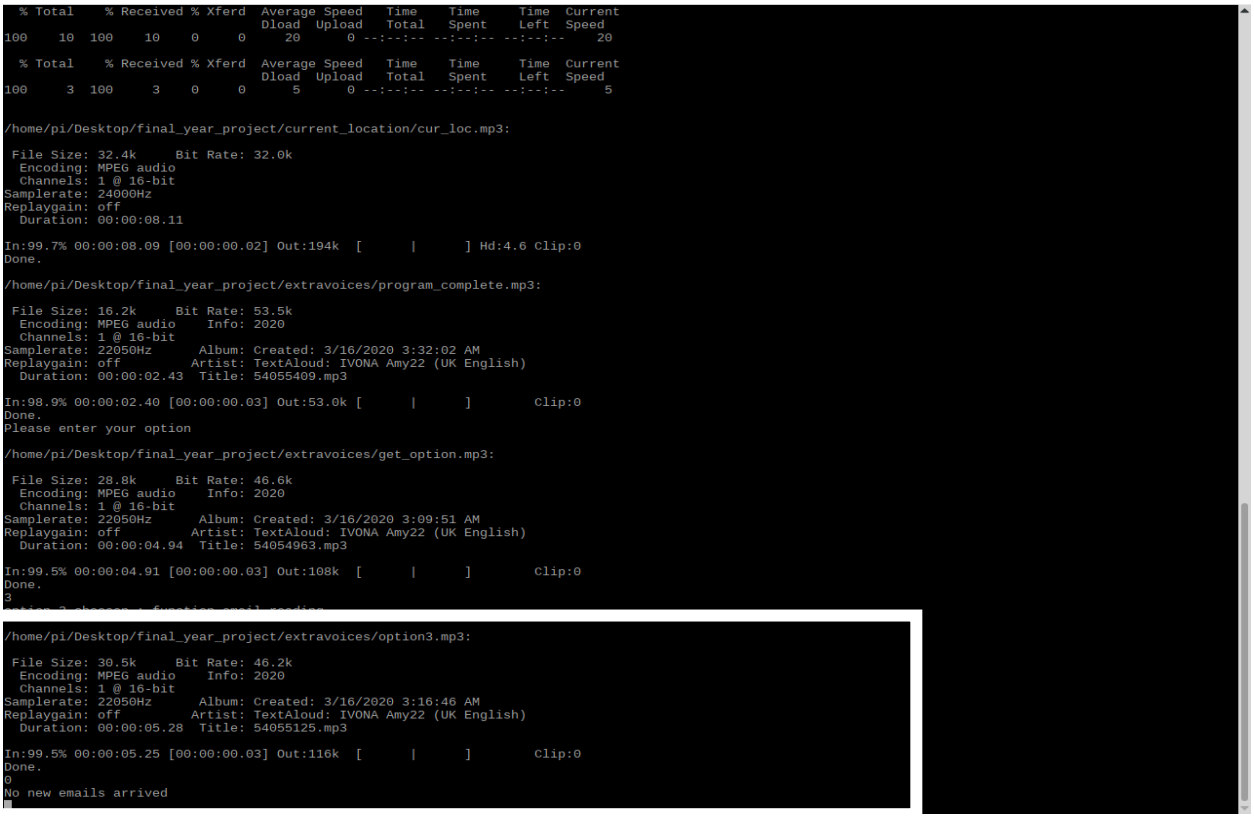
Experimental Results

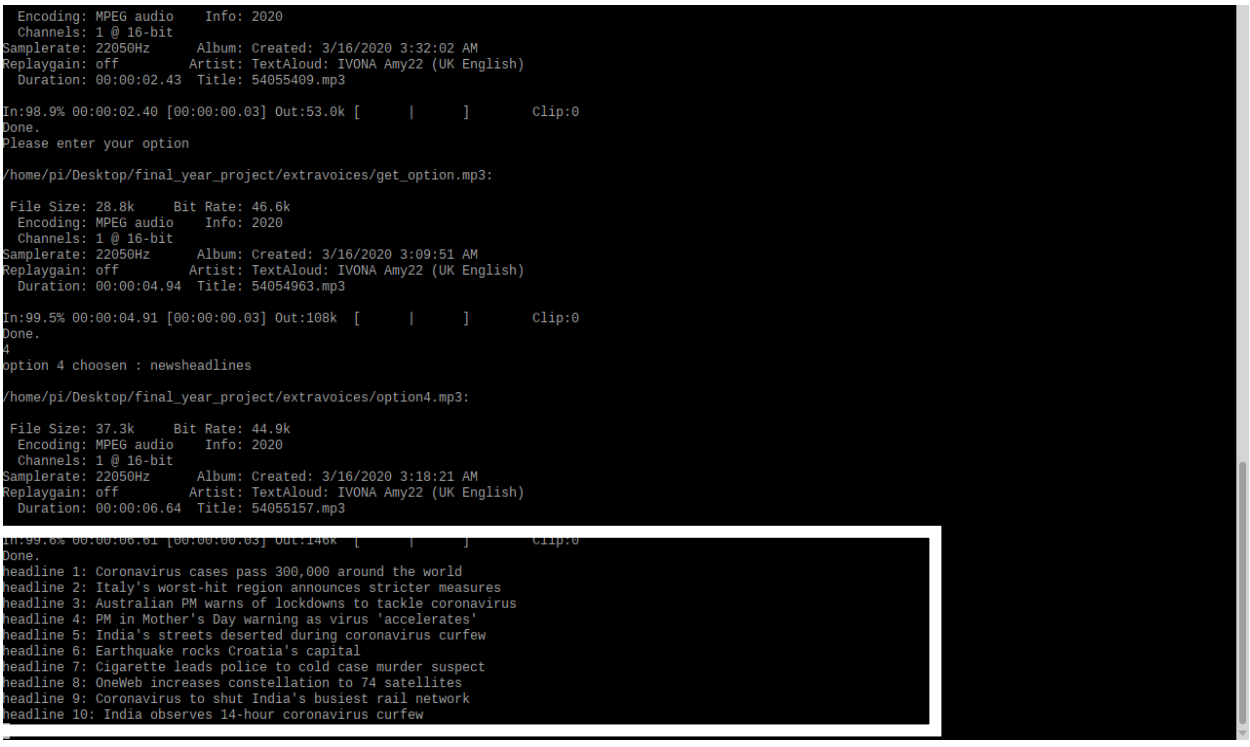
6.1 Outcome of Proposed System



```
/home/pi/Desktop/final_year_project/extravoices/program_complete.mp3:

 File Size: 16.2k     Bit Rate: 53.5k
  Encoding: MPEG audio    Info: 2020
  Channels: 1 @ 16-bit
Samplerate: 22050Hz      Album: Created: 3/16/2020 3:32:02 AM
Replaygain: off         Artist: TextAloud: IVONA Amy22 (UK English)
  Duration: 00:00:02.43  Title: 54055409.mp3

In:98.9% 00:00:02.40 [00:00:00.03] Out:53.0k [      |      ]        Clip:0
Done.
Please enter your option

/home/pi/Desktop/final_year_project/extravoices/get_option.mp3:

 File Size: 28.8k     Bit Rate: 46.6k
  Encoding: MPEG audio    Info: 2020
  Channels: 1 @ 16-bit
Samplerate: 22050Hz      Album: Created: 3/16/2020 3:09:51 AM
Replaygain: off         Artist: TextAloud: IVONA Amy22 (UK English)
  Duration: 00:00:04.94  Title: 54054963.mp3

In:99.5% 00:00:04.91 [00:00:00.03] Out:108k  [      |      ]        Clip:0
Done.
2
option 2 choosen : function current location

/home/pi/Desktop/final_year_project/extravoices/option2.mp3:

 File Size: 37.3k     Bit Rate: 44.9k
  Encoding: MPEG audio    Info: 2020
  Channels: 1 @ 16-bit
Samplerate: 22050Hz      Album: Created: 3/16/2020 3:14:51 AM
Replaygain: off         Artist: TextAloud: IVONA Amy22 (UK English)

In:99.6% 00:00:06.61 [00:00:00.03] Out:146k  [      |      ] Hd:3.4 Clip:0
Done.
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    10  100    10    0     0      1      0  0:00:10  0:00:05  0:00:05     2

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    10  100    10    0     0     20      0 --:--:-- --:--:-- --:--:--    20

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100     3  100     3    0     0      5      0 --:--:-- --:--:-- --:--:--     5

 File Size: 32.4k     Bit Rate: 32.0k
  Encoding: MPEG audio
  Channels: 1 @ 16-bit
Samplerate: 24000Hz
Replaygain: off
  Duration: 00:00:08.11

In:92.6% 00:00:07.51 [00:00:00.60] Out:180k  [-=====|=====-] Hd:4.6 Clip:0
```

```
pi@raspberrypi:~ $ cd /home/pi/Desktop/final_year_project/controller_shell
pi@raspberrypi:~/Desktop/final_year_project/controller_shell $ sh control.sh

/home/pi/Desktop/final_year_project/extravoices/welcome_speech.mp3:

 File Size: 91.1k     Bit Rate: 41.9k
  Encoding: MPEG audio    Info: 2020
  Channels: 1 @ 16-bit
Samplerate: 22050Hz      Album: Created: 3/16/2020 2:45:43 AM
Replaygain: off         Artist: TextAloud: IVONA Amy22 (UK English)
  Duration: 00:00:17.40  Title: 54054527.mp3

In:99.8% 00:00:17.37 [00:00:00.03] Out:383k  [      |      ]        Clip:0
Done.
Please enter your option

/home/pi/Desktop/final_year_project/extravoices/get_option.mp3:

 File Size: 28.8k     Bit Rate: 46.6k
  Encoding: MPEG audio    Info: 2020
  Channels: 1 @ 16-bit
Samplerate: 22050Hz      Album: Created: 3/16/2020 3:09:51 AM
Replaygain: off         Artist: TextAloud: IVONA Amy22 (UK English)
  Duration: 00:00:04.94  Title: 54054963.mp3

In:99.5% 00:00:04.91 [00:00:00.03] Out:108k  [      |      ]        Clip:0
Done.
7
option 7 choosen : function current weather

/home/pi/Desktop/final_year_project/extravoices/temperature.mp3:

 File Size: 32.4k     Bit Rate: 45.8k
  Encoding: MPEG audio    Info: 2020
  Channels: 1 @ 16-bit
Samplerate: 22050Hz      Album: Created: 3/16/2020 10:09:30 AM
Replaygain: off         Artist: TextAloud: IVONA Amy22 (UK English)
  Duration: 00:00:05.67  Title: 54064285.mp3

In:99.5% 00:00:05.64 [00:00:00.03] Out:124k  [      |      ]        Clip:0
Done.
Temperature (in kelvin unit) = 304.34
atmospheric pressure (in hPa unit) = 1014
humidity (in percentage) = 42
description = few clouds
```

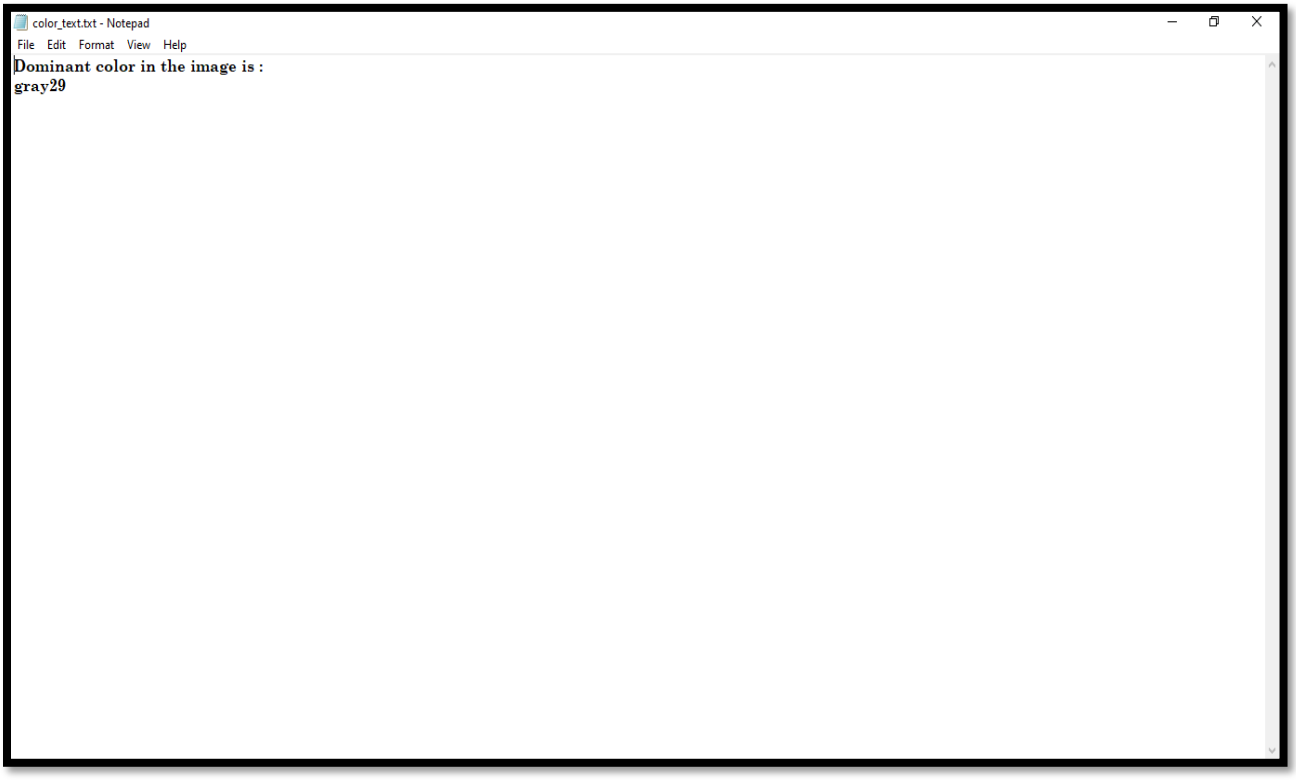

% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 10 100 10 0 0 20 0 --:--:-- --:--:-- --:--:-- 20
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 3 100 3 0 0 5 0 --:--:-- --:--:-- --:--:-- 5
/home/pi/Desktop/final_year_project/current_location/cur_loc.mp3:
File Size: 32.4k Bit Rate: 32.0k
Encoding: MPEG audio
Channels: 1 @ 16-bit
Samplerate: 24000Hz
Replaygain: off
Duration: 00:00:08.11
In:99.7% 00:00:08.09 [00:00:00.02] Out:194k [|] Hd:4.6 Clip:0
Done.
/home/pi/Desktop/final_year_project/extravoices/program_complete.mp3:
File Size: 16.2k Bit Rate: 53.5k
Encoding: MPEG audio Info: 2020
Channels: 1 @ 16-bit
Samplerate: 22050Hz Album: Created: 3/16/2020 3:32:02 AM
Replaygain: off Artist: TextAloud: IVONA Amy22 (UK English)
Duration: 00:00:02.43 Title: 54055409.mp3
In:98.9% 00:00:02.40 [00:00:00.03] Out:53.0k [|] Clip:0
Done.
Please enter your option
/home/pi/Desktop/final_year_project/extravoices/get_option.mp3:
File Size: 28.8k Bit Rate: 46.6k
Encoding: MPEG audio Info: 2020
Channels: 1 @ 16-bit
Samplerate: 22050Hz Album: Created: 3/16/2020 3:09:51 AM
Replaygain: off Artist: TextAloud: IVONA Amy22 (UK English)
Duration: 00:00:04.94 Title: 54054963.mp3
In:99.5% 00:00:04.91 [00:00:00.03] Out:108k [|] Clip:0
Done.
3
/home/pi/Desktop/final_year_project/extravoices/option3.mp3:
File Size: 30.5k Bit Rate: 46.2k
Encoding: MPEG audio Info: 2020
Channels: 1 @ 16-bit
Samplerate: 22050Hz Album: Created: 3/16/2020 3:16:46 AM
Replaygain: off Artist: TextAloud: IVONA Amy22 (UK English)
Duration: 00:00:05.28 Title: 54055125.mp3
In:99.5% 00:00:05.25 [00:00:00.03] Out:116k [|] Clip:0
Done.
0
No new emails arrived


Encoding: MPEG audio Info: 2020
Channels: 1 @ 16-bit
Samplerate: 22050Hz Album: Created: 3/16/2020 3:32:02 AM
Replaygain: off Artist: TextAloud: IVONA Amy22 (UK English)
Duration: 00:00:02.43 Title: 54055409.mp3
In:98.9% 00:00:02.40 [00:00:00.03] Out:53.0k [|] Clip:0
Done.
Please enter your option
/home/pi/Desktop/final_year_project/extravoices/get_option.mp3:
File Size: 28.8k Bit Rate: 46.6k
Encoding: MPEG audio Info: 2020
Channels: 1 @ 16-bit
Samplerate: 22050Hz Album: Created: 3/16/2020 3:09:51 AM
Replaygain: off Artist: TextAloud: IVONA Amy22 (UK English)
Duration: 00:00:04.94 Title: 54054963.mp3
In:99.5% 00:00:04.91 [00:00:00.03] Out:108k [|] Clip:0
Done.
4
option 4 choosen : newsheadlines
/home/pi/Desktop/final_year_project/extravoices/option4.mp3:
File Size: 37.3k Bit Rate: 44.9k
Encoding: MPEG audio Info: 2020
Channels: 1 @ 16-bit
Samplerate: 22050Hz Album: Created: 3/16/2020 3:18:21 AM
Replaygain: off Artist: TextAloud: IVONA Amy22 (UK English)
Duration: 00:00:06.64 Title: 54055157.mp3
Done.
headline 1: Coronavirus cases pass 300,000 around the world
headline 2: Italy's worst-hit region announces stricter measures
headline 3: Australian PM warns of lockdowns to tackle coronavirus
headline 4: PM in Mother's Day warning as virus 'accelerates'
headline 5: India's streets deserted during coronavirus curfew
headline 6: Earthquake rocks Croatia's capital
headline 7: Cigarette leads police to cold case murder suspect
headline 8: OneWeb increases constellation to 74 satellites
headline 9: Coronavirus to shut India's busiest rail network
headline 10: India observes 14-hour coronavirus curfew

OCR feature


```
Raspberry Pi 3Program complete, complete
```

Color detection feature



Chapter 7

Testing

7.1 Testing and Validations

Software testing is the process used to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not and to identify the defects to ensure that the product is defect free in order to produce the quality product. Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment.

In general, the following properties indicate the extent to which the component or system under test:

- Meets the requirements that guided its design and development,
- Responds correctly to all kinds of inputs,
- Performs its functions within an acceptable time,
- Is sufficiently usable,
- Can be installed and run in its intended environments, and
- Achieves the general result its stakeholders desire.

Software testing can be conducted as soon as executable software (even if partially complete) exists. The overall approach to software development often determines when and how testing is conducted.

7.2 Testing Levels

A level of software testing is a process where every unit or component of a software/system is tested. The main goal of system testing is to evaluate the system's compliance with the specified needs.

There are many different testing levels which help to check behavior and performance for software testing. These testing levels are designed to recognize missing areas and reconciliation between the development lifecycle states. In SDLC models there are characterized phases such as requirement gathering, analysis, design, coding or execution, testing, and deployment. All these phases go through the process of software testing levels.

7.2.1 Functional Testing

FUNCTIONAL TESTING is a type of software testing whereby the system is tested against the functional requirements/specifications. Functions (or features) are tested by feeding them input and examining the output. Functional testing ensures that the requirements are properly satisfied by the application.

Functional testing types include:

- Unit testing.
- Integration testing.
- System testing.
- Sanity testing.
- Smoke testing.
- Interface testing.
- Regression testing.
- Beta/Acceptance testing.

7.2.2 Non-Functional Testing

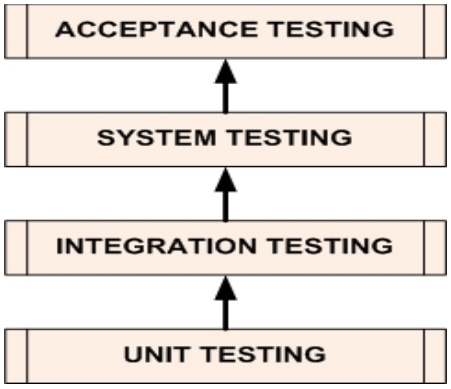
Non-functional testing is defined as a type of Software testing to check non-functional aspects (performance, usability, reliability, etc.) of a software application. It is designed to test the readiness of a system as per nonfunctional parameters which are never addressed by functional testing.

Examples of non-functional tests include:

- Load/Performance testing.
- Compatibility testing.
- Localization testing.
- Security testing.
- Reliability testing.
- Stress testing.
- Usability testing.
- Compliance testing.

7.3 Whitebox Testing

White Box Testing (also known as Clear Box Testing, Open Box Testing, Glass Box Testing, Transparent Box Testing, Code-Based Testing or Structural Testing) is a software testing method in which the internal structure/design of the item being tested is known to the tester.



Stage – 1: Unit Testing

This is the stage where the developers dissect the software and scrutinize its smallest units to find out any grass root level problems. Here the focus is on analyzing and testing each and every unit of every module to see whether it's working properly.

Stage – 2: Integration Testing

Moving on from the units, the next stage involves testing how well the various modules and components are integrated within the developed software. The integration is checked both ways that is top-down as well as bottom-up, so as to bring out the design, construction and architectural defects in the software. It's at this stage that most of the basic design flaws of the software will become obvious. The various interfaces will also be tested for defects at this stage.

Stage – 3: Sub-System and System Testing

This stage focuses on validating and analyzing that the software and all its sub-systems comply with the requirements as specified by the client. It's at this stage the software is tested as a whole.

Stage – 4: Testing Systems Engineering

The objective of this fourth stage of software testing is to see whether the software works well when integrated with external components like computer systems and other software, as specified in the software requirements provided by the end user or client. It's important to note here that the software will not be used on the developer's computer system, so testing must be made keeping in mind the computer system on which the software will ultimately be used.

Stage – 5: User Testing

This final stage is also known as acceptance testing stage, wherein the end user or some representative tests the final software to see if its complete and it actually performs the functions it is supposed to perform.

7.4.1 Unit Testing

Unit testing, a testing technique using which individual modules are tested to determine if there are any issues by the developer himself. It is concerned with functional correctness of the standalone modules. The main aim is to isolate each unit of the system to identify, analyze and fix the defects. Unit Testing has its advantages, some of them are:

- Reduces Defects in the newly developed features or reduces bugs when changing the existing functionality.
- Reduces Cost of Testing as defects are captured in very early phase.
- Improves design and allows better refactoring of code.
- Unit Tests, when integrated with build gives the quality of the build as well.

Test ID	Test Case Description	Input data	Expected Output	Actual Output	Status
1	Date and time	Date and time option selected	Current date and time	Current date and time	Pass
2	Current location	Current location option selected	Current city, region and country	Current city, region and country	Pass
3	Current weather	Current weather option selected	Current weather info	Current weather info	Pass
4	Email	Email option selected	Unread emails	Unread emails	Pass

5	News headlines	News headlines option selected	Top 10 news headlines	Top 10 news headlines	Pass
6	OCR	OCR option selected	Image to text to speech	Image to text to speech	Pass
7	Color detection	Color detection option selected	Identify dominant color in image	Identify dominant color in image	Pass

Table 7.1 Unit Testing

7.4.2 Integration Testing

In Integration Testing, individual software modules are integrated logically and tested as a group. A typical software project consists of multiple software modules, coded by different programmers. Integration testing focuses on checking data communication amongst these modules. Although each software module is unit tested, defects still exists for various reasons like:

- A Module in general is designed by an individual software developer whose understanding and programming logic may differ from other programmers.
- At the time of module development, there are wide chances of change in requirements by the clients. These new requirements may not be unit tested.
- Interfaces of the software modules with the database could be erroneous.

Test ID	Test Case Description	Input data	Expected Output	Actual Output	Status
1	Controller shell	Particular option	Execute corresponding program	Execute corresponding program	Pass
2	Date and time	Date and time option selected	Current date and time	Current date and time	Pass
3	Current location	Current location option selected	Current city, region and country	Current city, region and country	Pass
4	Current weather	Current weather option selected	Current weather info	Current weather info	Pass
5	Email	Email option selected	Unread emails	Unread emails	Pass

Table 7.2 Integration Testing

7.4.1 System Testing

System testing is performed on a completely integrated system which would meet the specified requirements. The main purpose of system testing is to check the problems between any of the software components. It takes all the components which have passed the integration testing. It not only checks the functionalities along with the behavior of the system. Some of the tests included in system testing are graphical user interface testing and usability testing.

Finally The system is tested altogether and all the subsystems that were tested are put together and tested as a whole system. The various components of the system work well together and it passes the test cases. This ensures that the user can easily and efficiently use the system.

Chapter 8

Conclusion and Future Enhancement

8.1 Conclusion

Technology today has grown to an extent where a machine is able to perform complex tasks by using machine learning. Machine learning and computer vision helps the machine to even retrieve meaningful data and information. Machine learning provides solution to complex problems and it is set to be pillar of our future civilization.

Smart glasses for visually impaired people is a project that uses such machine learning models to provide a assistive device for the visually impaired user. The developed product helps the visually impaired user to know the current date and time, current weather information, current location, get unread emails, get the text in an image, get the dominant color in an image, get the top 10 news headlines and also provide obstacle avoidance capacity. All the features have voice-based output to the visually impaired user. These features are accessible to the visually impaired user through the push of single particular button on the hand held remote. The processing is done by the Raspberry pi processor. The final product developed is easy to use and provides efficient results.

This project can be enhanced in the future to add more features and tide over the gap between the visually impaired people with rest of the society.

8.2 Future Enhancement

The processing of all the features here in the developed product is performed by the raspberry pi processor. For further enhancements of the project we can send the data form the webcam and remote to the cloud where the processing is made and the results are sent out to the audio output devices. This enhancement would eradicate the use of raspberry pi in the product and thus would make the product much lighter and easier to use for the visually impaired user. Even advancements of the project we may integrate real time object detection feature which would voice out all the objects surrounding the visually impaired user. Various other features like blind navigation can also be added to the product.