The Islamic Universityof Gaza
Faculty of Information Technology
Department of Multimedia& Web Development

الجامعـــــــــة الإســـــلاميــة بغـزة
كلية تكنولوجيا المعلومات
قسم وسائط ومتعددة وتطوير الويب

Peace Ear: Smart Glass for Deaf People

By

Nadia Zuhair Khatib

Supervised by
Dr.Motaz Saed

A Graduation Project Report Submitted to the
Faculty of Information Technology
In Partial Fulfillment for the degree of Bachelor in Multimedia and Web Development
The Islamic University Of Gaza

August /2020

Abstract

Individuals who are Deaf or hard of hearing can experience trouble completing basic daily tasks such as interacting with others. Although some solutions are used to deal with this, they have a lot of flaws. Where lip reading can be used, but only to understand about 30% of the spoken words, although there are sign language interpreters, they are few, and this deficiency is expected to worsen as these individuals enter public situations more often. These problems are partly responsible for the high unemployment rate and the existence of problems such as depression and anxiety in the deaf community.In this project, we design smart glasses that are assistive technology for people with hearing disabilities to help us understand what is going around them and the possibility of coexistence and interaction with the other worlds. The glasses are able to provide accurate real-time speech transcription and properly format it onto its display. When worn, they are attached to a regular pair of glasses, where are compatible with any eyeglass prescription the wearer may have. Overall, the device is highly effective and is able to achieve its purpose. where The program detects when a person is speaking by analyzing the energy levels (the sound intensity) of the stream of audio coming from the microphone.and then the program is analyzing the audio from the microphone, it is changed based on the current level of ambient noise. This allows the program to adapt to the situation the wearer of the glasses is in. We use the prototyping methodology to develop this project because the product requirements are not clearly understood, unstable, and change rapidly. This project will have a major impact on people with hearing disabilities, as it will help them to communicate effectively with friends and members of society, contribute to raising their awareness and improving their educational and cultural level, and also works to increase

their productivity and participation in all fields, and helps encourage community members to speak with people with hearing disabilities and participate in with them.

أذن السلام: نظارة ذكية للصم

ملخص الدراسة

يواجه الأفراد الصم أو ضعاف السمع مشكلة في إكمال المهام اليومية الأساسية مثل التفاعل مع الآخرين. حيث يتم استخدام الحلول للتعامل مع هذا ولكنها بها عيوب ؛ يمكن استخدام قراءة الشفاه فقط لفهم حوالي 30٪ من الكلمات المنطوقة ، وهناك نقص في مترجمي لغة الإشارة من المتوقع أن يتفاقم مع دخول هؤلاء الأفراد إلى المواقف العامة في كثير من الأحيان. حيث تعود هذه المشاكل جزئيًا إلى ارتفاع معدل البطالة ووجود مشاكل مثل الاكتئاب والقلق في مجتمع الصم.

في هذا المشروع ، نصمم نظارات ذكية تساعد الأشخاص ذوي الإعاقات السمعية على فهم ما يدور حولهم وإمكانية التعايش والتفاعل مع العوالم الأخرى.

النظارات قادرة على توفير نسخ دقيق للكلام في الوقت الحقيقي وتنسيقه بشكل صحيح على الشاشة. عند ارتدائها ، يتم إرفاقها بزوج عادي من النظارات ، حيث تكون متوافقة مع أي وصفة طبية قد تكون لدى مرتديها. بشكل عام ، الجهاز فعال للغاية وقادر على تحقيق الغرض منه. حيث يكتشف البرنامج عندما يتحدث الشخص عن طريق تحليل مستويات الطاقة (شدة الصوت) لدفق الصوت القادم من الميكروفون ثم يقوم البرنامج بتحليل الصوت من الميكروفون ويتم تغييره بناءً على المستوى الحالي لـ الضوضاء المحيطة. يسمح هذا للبرنامج بالتكيف مع الوضع الذي يكون فيه مرتدي النظارات.

نحن نستخدم منهجية النماذج الأولية لتطوير هذا المشروع لأن متطلبات المنتج ليست مفهومة بشكل واضح وغير مستقرة وتتغير بسرعة.

سيكون لهذا المشروع تأثير كبير على الأشخاص ذوي الإعاقة السمعية ، حيث سيساعدهم على التواصل بشكل فعال مع الأصدقاء وأفراد المجتمع ، ويساهم في رفع وعيهم وتحسين مستواهم التعليمي والثقافي ، كما يعمل على زيادة إنتاجيتهم ومشاركتهم في جميع المجالات ، ويساعد في تشجيع أفراد المجتمع على التحدث مع الأشخاص ذوي الإعاقة السمعية والمشاركة معهم.

بسم الله الرحمن الرحيم

Acknowledgment

All Thanks
To Allah
Who gave me the ability to successfully complete the graduation project

All Thanks
To our parents
Who gave me great support and help to reach this achievement

All Thanks
To our teachers
Especially Our supervisor, **Dr. Motaz Saad**
For his efforts with me during the project

All Thanks
To everyone who encouraged me to successfully finish
the graduation project

Table of Contents

List of Figures

List of Abbreviations

ALSA	Advanced Linux Sound Architecture
API	Application Programming Interface
AR	Augmentad Reality
GPIO	General Purpose Input/Output
I2C	Inter-Integrated Circuit
LEDS	Light Emitting Diodes
LIPO	Lithium Polymer
LTCCS	The Live Time Closed Captioning System
NOOBS	New Out Of Box Software
OLED	Organic Light Emitting Diodes
OS	Operating System
RDP	Remote Desktop Program
RPI	Rasbperry Pi
SBC	Single-Board Computer
SD CARD	Security Digital Card
SPI	Serial Peripheral Interface
SSH	Secure Shell
STT	Speech To Text
VNC	Virtual Network Computing
Win32	Window 32

Chapter 1

Introduction

1.1 Introduction and Background

Could you imagine how the life of a person who is deaf could be?
Many of them can't understand what happens without the help of sign language interpreters.

Their life always depends upon their sign language interpreters, lipreading can only be used to understand about 30% of the words being spoken and can be quite difficult for them alone.

The increasing number of people with disabilities in the world attracts the concern of researchers to invent various technologies, hoping that these technologies can assist the disabled people in carrying out their tasks in everyday life like normal people.
So I want to make something for them that would help them become independent. A smart glasses project is what I want to create.
(**Figure 1, 2**: Show the number of deaf people in some countries especially Palestine[in 2017]) [2] [1]

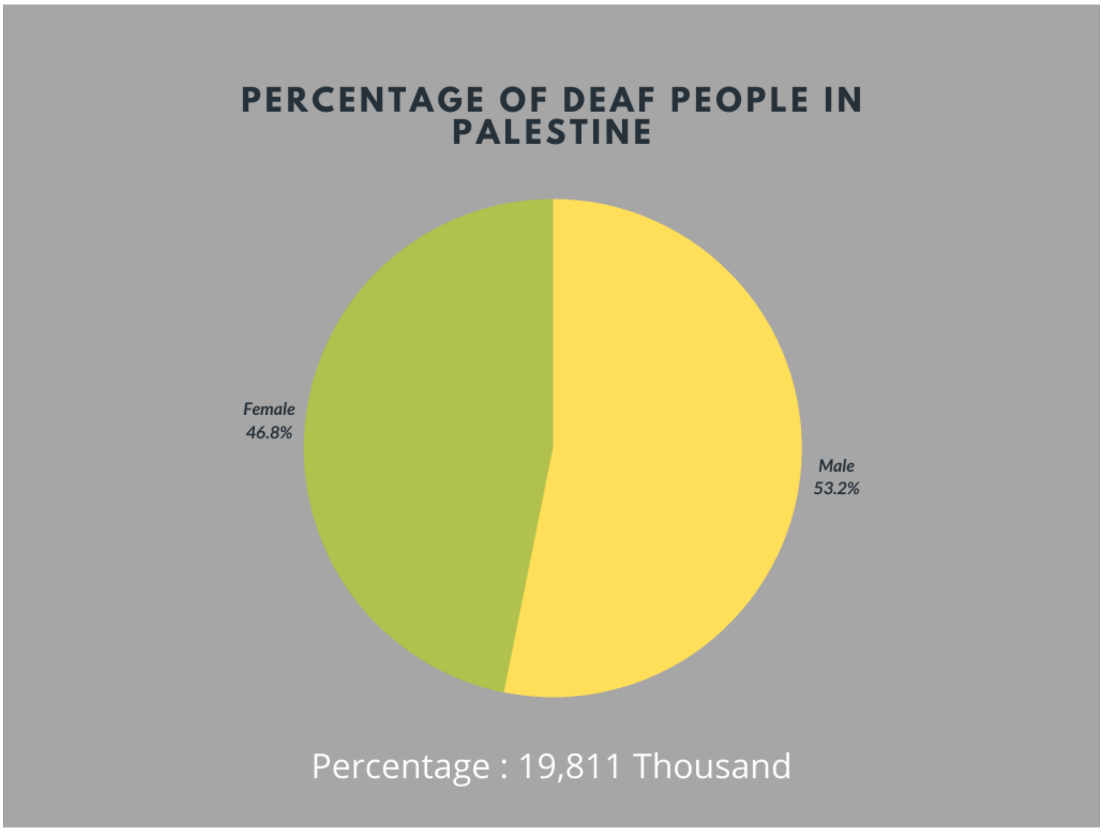

Figure 1: Deaf People in Palestine

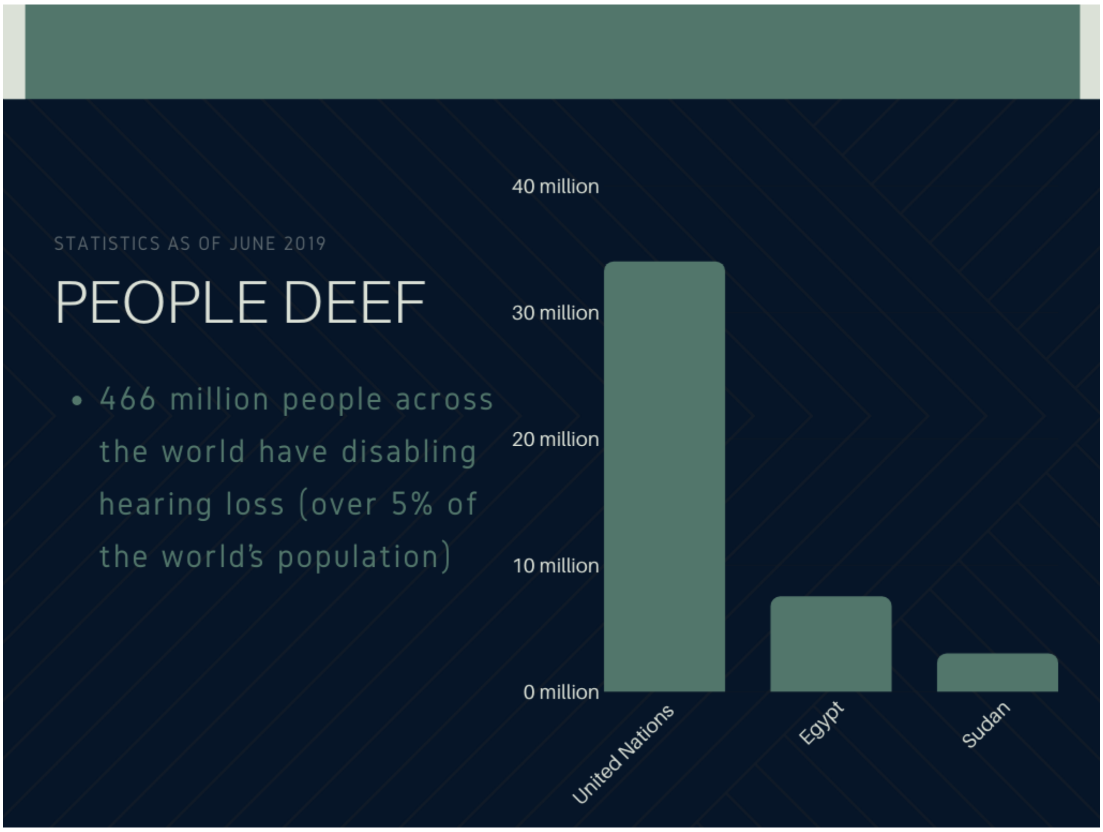

Figure 2: Deaf People in Some Countries

These smart glasses can assist them understand what is going on around them while walking alone in Any place in the community and in new environments by taking inputs audio through a Microphone and providing converted text to the person through the old display. So deaf people can be trained and helped to carry out their tasks in everyday life using wearable devices programmed.

Background:

- **Smart Glasses:**

Smart glasses are products that are worn like normal glasses. They provide the user with information and technological possibilities to take pictures or record video or capture audio. They use optics technology based on a Heads-Up Display (HUD), a Head-Mounted Display (HMD), and in particular, an Optical Head Mounted Display (OHMD). There is a plastic object at eye-level through which the user can see both an online, digital world and offline. The added information can be shown visually before the eyes through the display of the glasses, or the user can get instructions, notifications, and answers to questions in audio form. [3, 4]

- **Raspberry Pi**

It is a series of small single-board computers(SBC) developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries. The original model became far more popular than anticipated, selling outside its target market for uses such as robotics. It is now widely used even in research projects, such as for weather monitoring because of its low cost and portability. It does not include peripherals (such as keyboards and mice) or cases. However, some accessories have been included in several official and unofficial bundles. [5]

- **Raspbian Stretch OS:**

Raspberry Pi OS is a Debian-based operating system for Raspberry Pi. Since 2015 it has been officially provided by the Raspberry Pi Foundation as the primary operating system for the family of Raspberry Pi single-board computers. The original Raspbian OS was created by Mike Thompson and Peter Green as an independent project. The initial build was completed in June 2012. [6]

Previous Pi OS have been 32bit and based on Raspbian core, taking the name Raspbian. Since recent 64bit versions no longer use the Raspbian core, the name has been changed to Raspberry Pi OS for both 64bit and 32bit versions. As of 1 August 2020, the 64-bit version is a beta and is not suitable for general use.

Raspberry Pi OS is highly optimized for the Raspberry Pi lines of compact Single Board Computers with ARM CPUs. Raspberry Pi OS uses a modified LXDE as its desktop environment with the Openbox stacking window manager plus a new theme and few other changes. The distribution is shipped with a copy of the algebra program Wolfram Mathematica and a version of Minecraft called Minecraft Pi as well as a lightweight version of Chromium as of the latest version.

- **Python Coding Language:**

Python is a handy and relatively easy to learn programming language. It’s flexible to allow users to build a web application as well as an interface with hardware components connected to Raspberry Pi, which makes it the perfect language to start learning on the Raspberry Pi(Python with Raspberry 2018). [7]

The Python programming language started as a scripting language for Linux. Python programs are similar to shell scripts in that the files contain a series of commands that the computer executes from top to bottom. Like shell scripts, Python can automate tasks like batch renaming and moving large amounts of files. It can be used just like a command line with IDLE, Python’s REPL (read, eval, print, loop) and others.

Python is a wonderful and powerful programming language that's easy to use (easy to read and write) and, with Raspberry Pi, lets you connect your project to the real world. [8]

- **Speech Recognition :**

It is the ability of a machine or program to identify words and phrases in spoken language and convert them to a machine-readable format. Rudimentary speech recognition software has a limited vocabulary of words and phrases, and it may only identify these if they are spoken very clearly. More sophisticated software has the ability to accept natural speech.

Speech recognition works using algorithms through acoustic and language modeling. Acoustic modeling represents the relationship between linguistic units of speech and audio signals; language modeling matches sounds with word sequences to help distinguish between words that sound similar. [9]

1.2 Statement of the Problem

- Individuals who are Deaf or hard of hearing can experience trouble completing basic daily tasks such as interacting with others. Although some solutions are used to deal with this, they have a lot of flaws. Where lip reading can be used, but only to understand about 30% of the spoken words, although there are sign language interpreters, they are few, and this deficiency is expected to worsen as these individuals enter public situations more often. These problems are partly responsible for the high unemployment rate and the existence of problems such as depression and anxiety in the deaf community.

1.3 Objective of the work

1.3.1 Main objectives:

- The main objective is to developed an attachment to a pair of glasses that provides a real-time transcription of the words being spoken so that it allows Deaf people to ability to engage in a naturally flowing conversation, which can restore relationships with close friends and family members.

1.3.2 Specific objectives:

- Understand the work of Speech Recognition API.
- Understanding the work of OLED Display.
- Connect and work a mic and Oled display With Glasses.

1.4 Importance of Project

1. Help people with hearing disabilities communicate more effectively with friends and community members.
2. Provide an appropriate and encouraging environment in which people with hearing disabilities can easily adapt to society.
3. Change how deaf people engage in day to day.
4. Increase the productivity and participation of Deaf people in all areas of society.
5. This project helps encourage people to talk to and share with Deaf people.
6. This project helps raise awareness and improve educational, cultural, and intellectual levels because they understand everything that is will happen around them.

1.5 Scope and limitations of the work

1.5.1 Scope:

a. The prototype (Smart Glasses) works on RPI Zero and will connect it to an OLED screen to display text on it and a small microphone to capture the speaker's voice.
b. This prototype (Smart Glasses) will be used in all fields.
c. The users of this prototype (smart glasses) are the group of people with hearing disabilities of all ages in the world, who speak English.

1.5.2 Limitations:

1. Not an available Raspberry Zero that appropriate to the project size, so I used Raspberry Pi 3 due to its availability now.
2. Dealing with wireless networks.
3. Continuous Power cuts during work then this leads to damage to the memory that contains the project.
4. Few availabilities of libraries that convert from speech to text, specifically that support the Arabic language and that can be used in our country.
5. Not an available of OLED display causes us to pay careful attention to any possible errors on this the OLED that may lead to damage.
6. Problems through display text in any language especially the Arabic language on the OLED display with python.
7. Errors during the 3D printing process of the model causing it to be returned and taken a lot of time.
8. The lack of suitable and reflector lenses makes us careful when cutting them to a suitable size and shape.

Chapter 2

Relatd Works

2.1 The Live Time Closed Captioning System

The LTCCS is a revolution in assistive technology for the hearing impaired. The LTCCS is an on-head wearable display that provides closed captioning for real-life events as they happen live. [10]

Advantages:

- Easy to Use.
- Fine-Tuned Microphonea.
- Optimized for natural conversation.

Disadvantages:

- Although the advantages of the prototype, the final model is not launched because the LTCCS Project officially shut down, has been unable to secure funding and deliver the promised product.

Figure 3: The Live Closed Captioning System

2.2 National Theatre uses Moverio

The Moverio BT-350 AR smart glasses are the latest in Epson’s line of technological products that use the power of augmented reality to transform everyday experiences that are often taken for granted, in this setting enhancing the entertainment experience for those with hearing loss. [11]

Advantages:

- Easy on the eyes and comfortable to see.
- The user is not cut from the scene and what they came to see, rather they will enjoy a more accessible and exciting experience.

Disadvantages:

- Just made it for hearing-impaired theatregoers to view the actors' performances.
- Just available to book for any performance in all three National Theatre stages

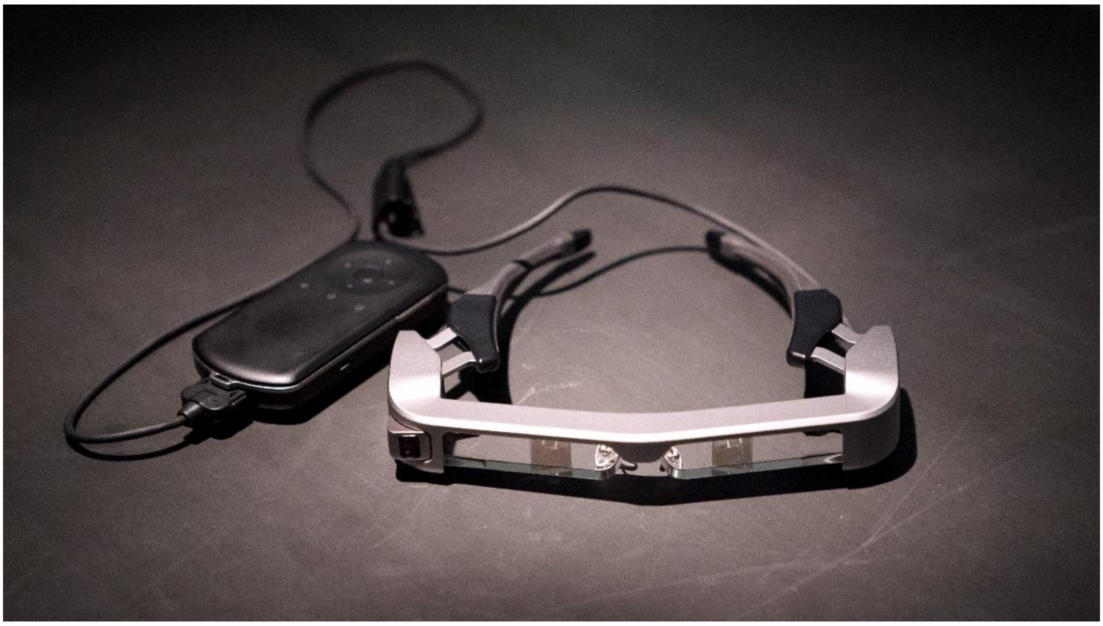

Figure 4: National Theatre Uses Moverio

2.3 Smart Glasses Light Drive

The Bosch Smartglasses Light Drive provides a complete, ready-to-use solution for smaller, lighter, more stylish smartglasses. The solution is also compatible with curved and corrective lenses. This all-in-one smartglasses system delivers a highly intuitive user experience as a convenient display extension, providing an alternative to smartphone or smartwatch screen checking. The light drive is composed of a technology stack of Bosch optical, computing, and sensor components. The optical systems include microelectromechanical system (MEMS) mirrors with ASIC controller that precisely direct a low-power beam of light to the user's eye.

Advantages:

- This Project Suitable for car drivers and visible for the eyes, invisible to others but Peace Ear suitable deaf people.
- This Project that takes image input but Peace Ear takes speech input.

Figure 5: Smart Glasses Light Drive

2.4 Summery between our project and related work

All of these works tried to help the deaf community with one thing or another and provide them with technology appropriate solutions, but there are some differences.

1. This project that serves deaf people directly in everything.
2. This project helps the deaf people in the world who speak English.
3. This project can clip on to any glasses, easy to use and comfortable.

Chapter 3

Methodology

3.1 Methodology

We will use prototyping methodology to develop this project because the requirements of the product are not clearly understood and are unstable or will changing quickly. [12]

Also, Benefits of prototyping include:

- Missing functionality and errors are detected easily.
- Prototypes can be reused in the future, more complicated projects.
- A system which is finished more quickly
- Less maintenance required
- Greater client satisfaction

During the project life cycle, we proceed in six stages as follows:
(**Figure 5**: Life Cycle)

Phase 1: Requirements gathering and analysis:

In this phase, the requirements of the system are defined in detail. During the process, the users of the system are interviewed to know what is their expectation from the system.

Phase 2: Quick design:

In this stage, a simple design of the system is created. However, it is not a complete design. It gives a brief idea of the system to the user. The quick design helps in developing the prototype.

Phase 3: Build a Prototype:

In this phase, an actual prototype is designed based on the information gathered from quick design. It is a small working model of the required system.

Phase 4: Initial user evaluation:

In this stage, the proposed system is presented to the client for an initial evaluation. It helps to find out the strength and weakness of the working model. Comment and suggestion are collected from the customer and provided to the developer.

Phase 5: Refining prototype:

In this stage, a simple design of the system is created. However, it is not a complete design. It gives a brief idea of the system to the user. The quick design helps in developing the prototype.

Phase 6: Implement Product and Maintain:

Once the final system is developed based on the final prototype, it is thoroughly tested and deployed to production. The system undergoes routine maintenance for minimizing downtime and prevent large-scale failures.

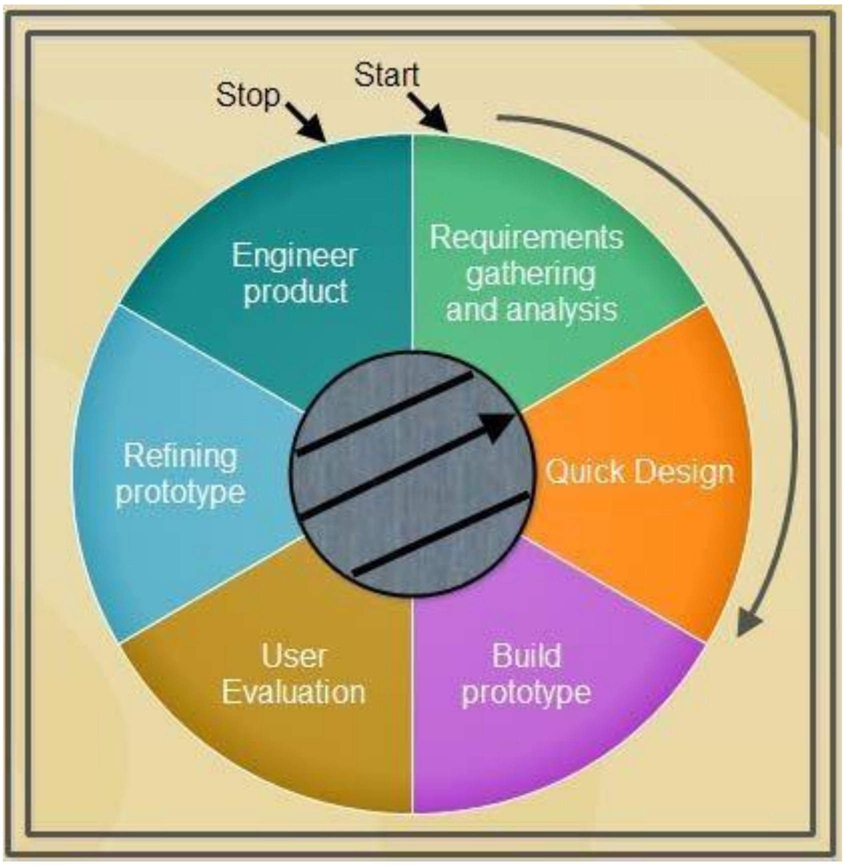

Figure 6: Prototyping Model Life Cycle

3.2 Team Member

- Nadia Khatib.

3.3 Project Schedualing and Time Table:

3.3.1 Time Table:

The Time Table of this project based on Software Development Methodology.

(**Figure 6-9**: Time Table)

Task Name	Duration	Start	Finish
Smart Glasses	202 days	ج ٠١/١١/١٩	ج ٠٧/٠٨/٢٠
Requirements gathering and analysis	42 days	ج ٠١/١١/١٩	إ ٣٠/١٢/١٩
Quick Design	12 days	ث ٣١/١٢/١٩	أ ١٥/٠١/٢٠
Build a Prototype	120 days	خ ١٦/٠١/٢٠	ث ٣٠/٠٦/٢٠
Initial user evaluation	6 days	أ ٠١/٠٧/٢٠	أ ٠٨/٠٧/٢٠
Refining prototype	5 days	خ ٠٩/٠٧/٢٠	أ ١٥/٠٧/٢٠
Implement Project and Maintain	17 days	خ ١٦/٠٧/٢٠	ج ٠٧/٠٨/٢٠

Figure 7: TimeTable 1.1

Smart Glasses	202 days	ج ٠١/١١/١٩	ج ٠٧/٠٨/٢٠
Requirements gathering and analysis	42 days	ج ٠١/١١/١٩	إ ٣٠/١٢/١٩
Requirements analysis	21 days	ج ٠١/١١/١٩	ج ٢٩/١١/١٩
Collect required requirements	22 days	س ٣٠/١١/١٩	إ ٣٠/١٢/١٩
Quick Design	12 days	ث ٣١/١٢/١٩	أ ١٥/٠١/٢٠
Preliminary sketch design	6 days	ث ٣١/١٢/١٩	ث ٠٧/٠١/٢٠
Prototype structure design	6 days	أ ٠٨/٠١/٢٠	أ ١٥/٠١/٢٠

Figure 8: TimeTable 1.2

Build a Prototype	120 days	خ ١٦/٠١/٢٠	ث ٣٠/٠٦/٢٠
Understand the requirements and relate them together	33 days	خ ١٦/٠١/٢٠	س ٢٩/٠٢/٢٠
Connect Speech Recognition,Oled Display	68 days	أ ٠١/٠٣/٢٠	إ ٠١/٠٦/٢٠
Simplified prototype work	10 days	ث ٠٢/٠٦/٢٠	إ ١٥/٠٦/٢٠
Install this prototype on glasses	11 days	ث ١٦/٠٦/٢٠	ث ٣٠/٠٦/٢٠

Figure 9: TimeTable 1.3

Initial user evaluation	**6 days**	أ ٠١/٠٧/٢٠	أ ٠٨/٠٧/٢٠
Test Prototype	3 days	أ ٠١/٠٧/٢٠	ج ٠٣/٠٧/٢٠
Taking comments and user feedback	4 days	س ٠٤/٠٧/٢٠	أ ٠٨/٠٧/٢٠
Refining prototype	**5 days**	خ ٠٩/٠٧/٢٠	أ ١٥/٠٧/٢٠
User feedback analysis	3 days	خ ٠٩/٠٧/٢٠	س ١١/٠٧/٢٠
Improved and added features on Prototype	4 days	أ ١٢/٠٧/٢٠	أ ١٥/٠٧/٢٠
Implement Project and Maintain	**17 days**	خ ١٦/٠٧/٢٠	ج ٠٧/٠٨/٢٠
Final prototype development	1 day	خ ١٦/٠٧/٢٠	خ ١٦/٠٧/٢٠
Final test	2 days	ج ١٧/٠٧/٢٠	س ١٨/٠٧/٢٠
Improvement and Maintenance	16 days	أ ١٩/٠٧/٢٠	ج ٠٧/٠٨/٢٠

Figure 10: TimeTable 1.4

3.3 Tools and Equipment:

3.3.1 Tools:

1. **Trello:**

   We use it to manage our project.

2. **Google Drive :**

   We use it to share the files.

3. **Google Docs**

   We use it to share the Documentation.

4. **Microsoft Office**

   -Word: we use it to write documentation.

   - PowerPoint: we use it to make a presentation for our project.

5. **Pycharm**

   I use the editor to write code.

6. **Github**

   I use it to keep version control to our system.

7. **BalenaEtcher**

   I used for writing image files such as .iso and .img files, as well as zipped folders onto storage media to create live SD cards and USB flash drives.

8. **Win 32 Disk**

A Windows tool for writing images to USB sticks or SD/CF cards

9. **Tinkercad Program**

   Online 3D modeling program that runs in a web browser to make design smart glasses.

3.3.2 Equipment:

(Figure 10: Equipment)

1. Laptop
2. A Tv or Computer Screen
3. An Ethernet Cable/ wireless
4. A USB Keyboard and Mouse
5. A Power Supply
6. Raspbeery Pi
7. MicroSD Card (Class 10) — 16GB
8. Lipo Battery
9. Oled Display
10. A USB Mini Mirophone
11. Small Mirror (for lens)
12. Clear Acrylic (Reflector for lens)
13. Lens
14. Pin GPIO Connector Header
15. Female to Female Jumper Cables
16. 3d Printer and filament

Figure 11: Equipments

3.4 Programming Languages and Libraries:

- Raspbian OS:

RPI OS is the recommended operating system for normal use on a Raspberry Pi.

- Python Language:

Used it to program Smart Glasses.

- Google Speech Recognition API:

Used it to convert speech to text.

- Luma.OLED:

I used it for show text from output file that converted onto OLED display.

Chapter 4
Design and Analysis

4.1 Requierment and Analysis

4.1.1 Functional Requirement: [13]

1. The main purpose of the project is to help deaf people to recognize the Audio in front of them through a set of steps:

- Glasses can capture Audio via Microphone.
- Convert Audio to text through Google Speech Recognition.
- Show the text on the old display to be understood by deaf people.
- Notify them if they No Audio around of them.

4.1.2 Non-Functional Requirement: [13]

- Performance:

    1. I use Raspberry Pi 3, with a quad-core ARM Cortex-A53 processor, which is described as having ten times the performance, faster and Easy to Mobility.
    2. (Focus on Software) using Google Speech Recognition or (Focus on Hardware) using Raspberry pi 3.
    3. The amount of data to be stored will be very high on memory it will not reflect on the performance of the project.

- Reliability:

    1. The project should be fully functional at least 95% of the time.
    2. In the event of a crash, the system should maintain full functionality after it is restored.

- Maintainability:

    1. The design of the project should be simple enough that they are easy to modify and easy to test.

- Usefulness:

1. It will be a great help for deaf users and it will make their life bit easier.

- Installation:

  1. It will be requirements in project easy to install/uninstall based on user needs and will take no time.

- Scalability:

  1. It will be able to be used by varying numbers of users, or with varying amounts of data.

- Portability:

  1. The glass is running using Raspberry pi 3 which can run using its power source, Microphone, and Oled display only without needing any device again.

4.1.3 Data Requirement:

- Types of inputs:

  a. Audio :
  The result of recognition is converted to text using Speech recognition in python using Google Speech API.

4.2 System Components

4.2.1 System Overview:

(Figure 12: System Overview**)**

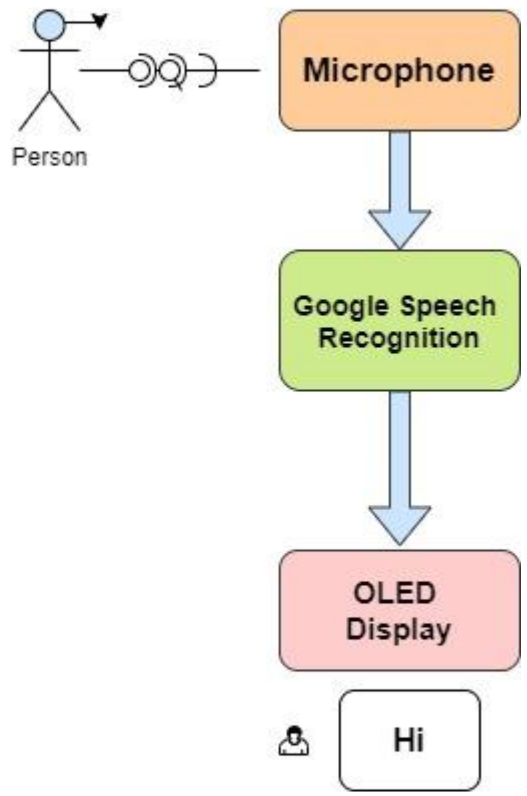

Figure 12: System Overview

- Design Prototype :

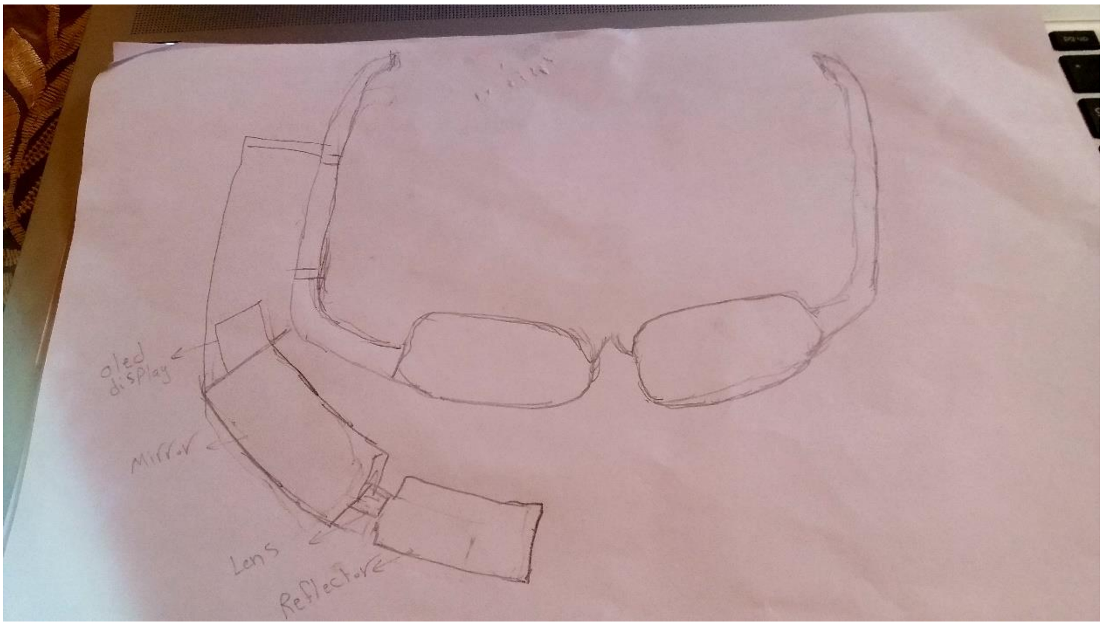

Figure 13: Initial Prototype

- Design Work Steps:

    - **Step 1**

    How it works:

    The device was programmed entirely in Python. There were some external libraries used in the programs, the most important being adafruit_ssd1306 Oled to control the display, SpeechRecognition to perform speech recognition. There are three parts to the main controlling program. First, the microphone starts recording audio until it detects that a phrase has been spoken. The audio is then sent to the Google Speech Recognition API, from which the word(s) spoken are received in the form of a String (a sequence of characters). These words are shown on the display, which allows the wearer of the glasses to see them.

    There are several variables that had to be adjusted to improve the quality and speed of word detection, including the energy threshold, dynamic energy adjustment ratio, pause threshold, and phrase time limit.

    Louder sounds result in energy levels which are higher. If the energy levels are below a certain threshold, known as the energy threshold, it is assumed that there is silence; if they are above the energy threshold, the audio is considered to be speech.

    The energy threshold is initialized to a certain value before the microphone is turned on and then the program is analyzing the audio from the microphone, it is changed based on the current level of ambient noise. This allows the program to adapt to the situation the wearer of the glasses is in.

    (Figure 14: Works)

Figure 14: How to Works

- **Step 2**

  The Lens:

  The lens was a tricky thing. I had not really an Idea about optics. So I googled, and I asked an optometrist about it. he suggested about it the best working lens is an Acrylic Plano-convex lens with a focal point of 100mm. If you place the screen at a distance of 73mm from the lens away, you will get a virtual image at a distance between 27-30cm. The magnification factor is x3.

That's perfect, because another important thing is, that your eyes can only start focusing things at +-25 cm.

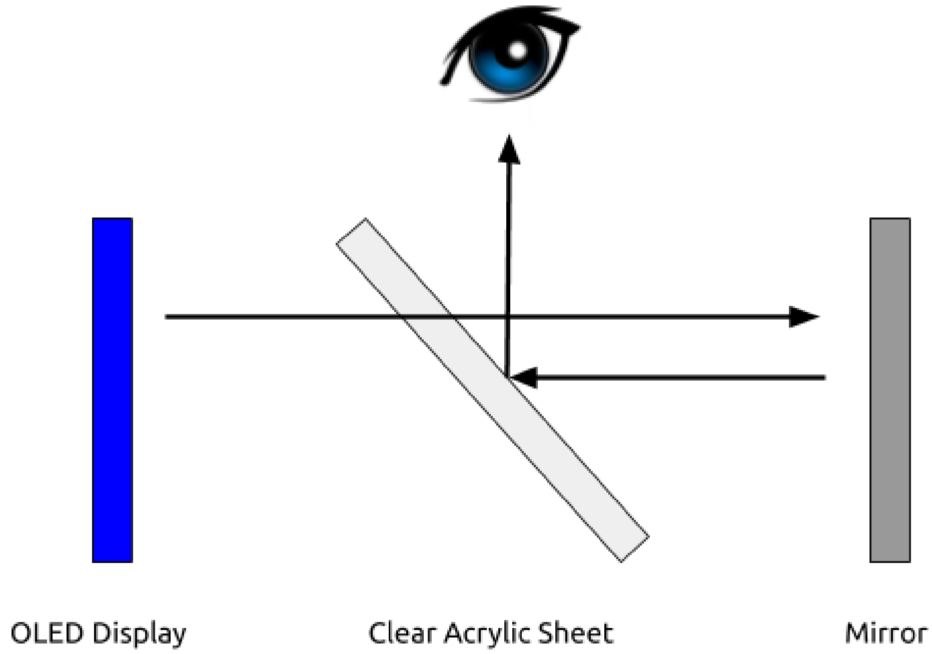

Figure 15: The Lens

- **Step 3**

The Reflector and Mirror:
(Figure 16: Mirror & Reflector)
These pieces were glued together using Amir industrial adhesive. It was chosen because of its high strength and ability to attach to non-porous substances such as the acrylic sheet that was used.
The dimensions of the pieces used to make the prism are given below:

- Mirror: 3.1cm X 0.9cm
- Clear Acrylic: 1.4cm X 4cm

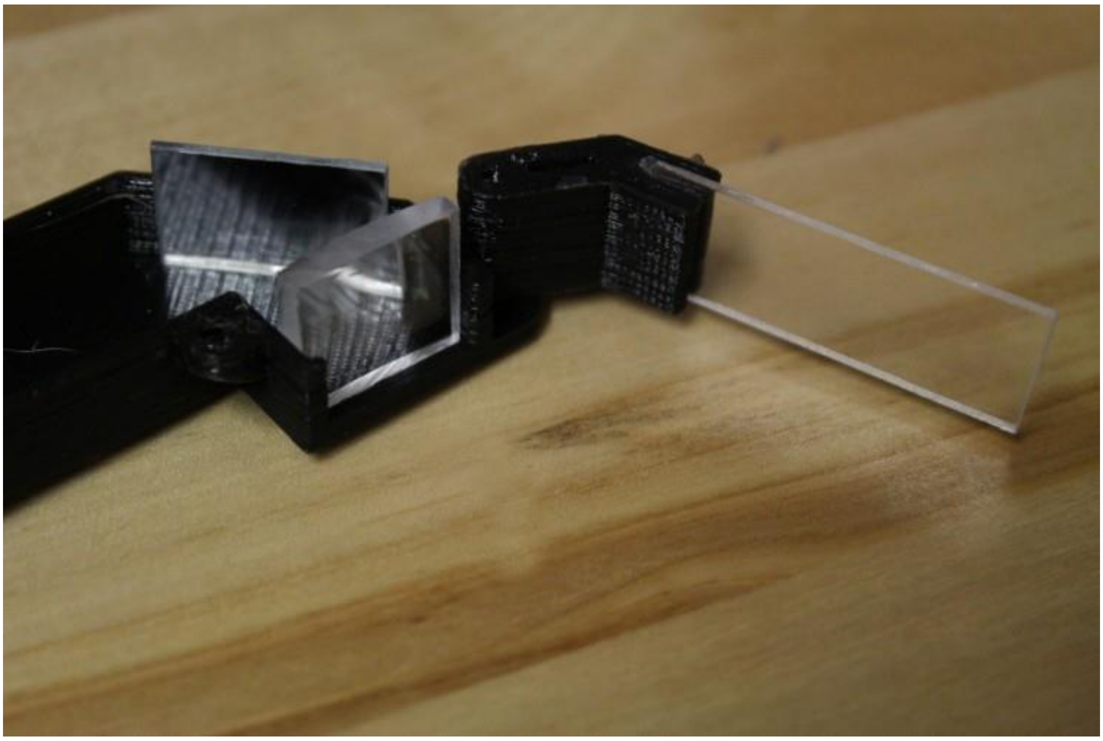

Figure 16: Mirror & Reflector

- **Step 4**

  The electronic:

  I used a 3D printer. I made a few modifications to the design to appropriate the prototype.

- **Step 5**

  (Figure 17: Connecting Oled)

  Connecting to Oled Display:

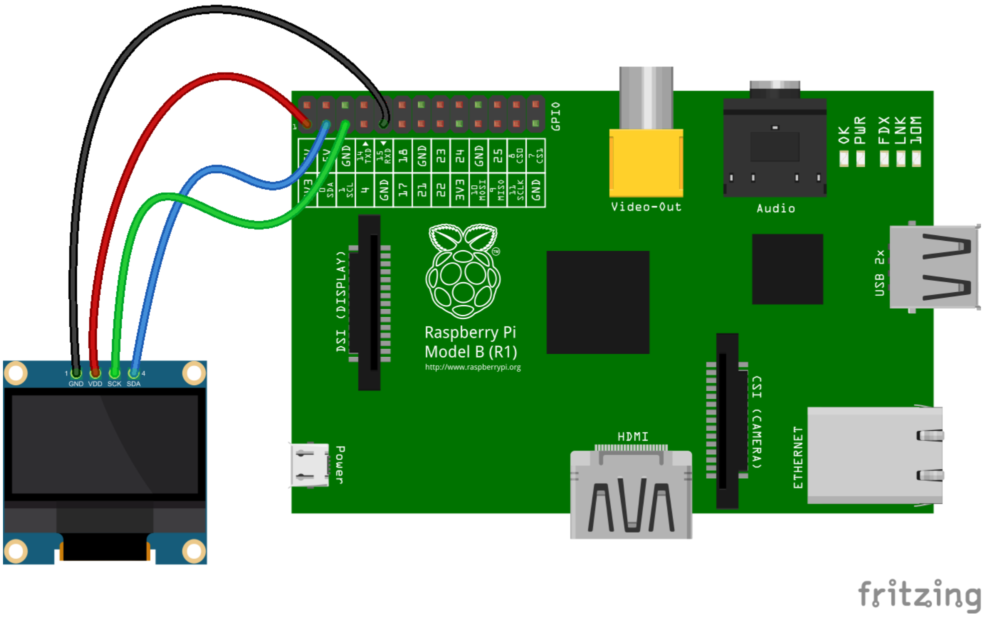

Figure 17: Connecting Oled

- **Step 6**

  Peace Ear Smart Glasses

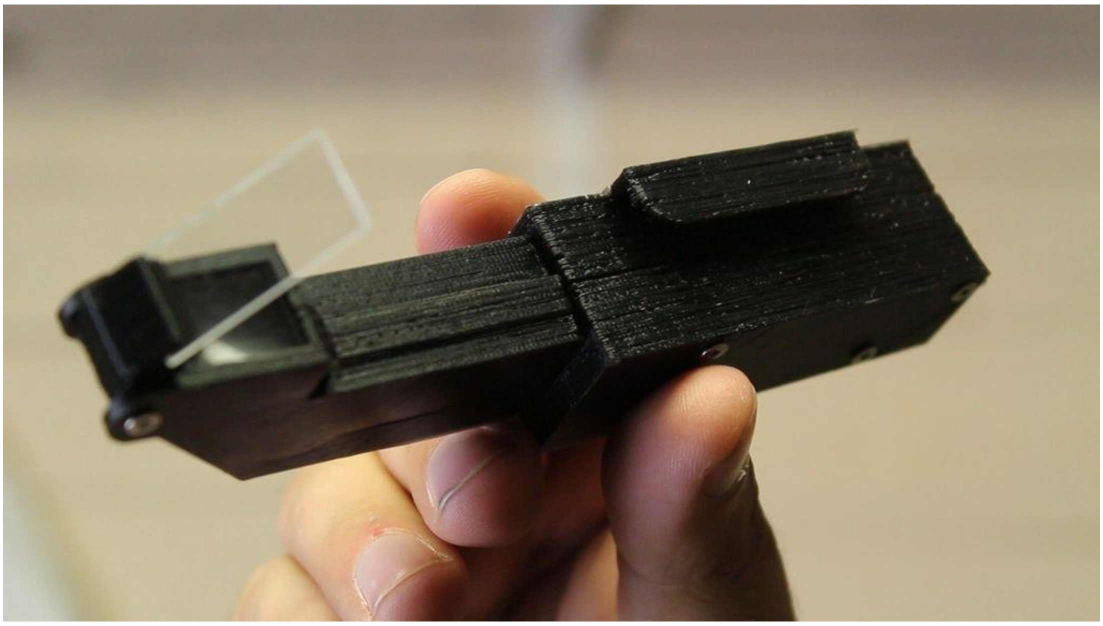

Figure 18: Peace Ear Smart Glasses

Chapter 5

Implementation

5.1 Meet Raspberry Pi

I started to know about RPI by the diagram, what things I need to use it, and how to set it up.

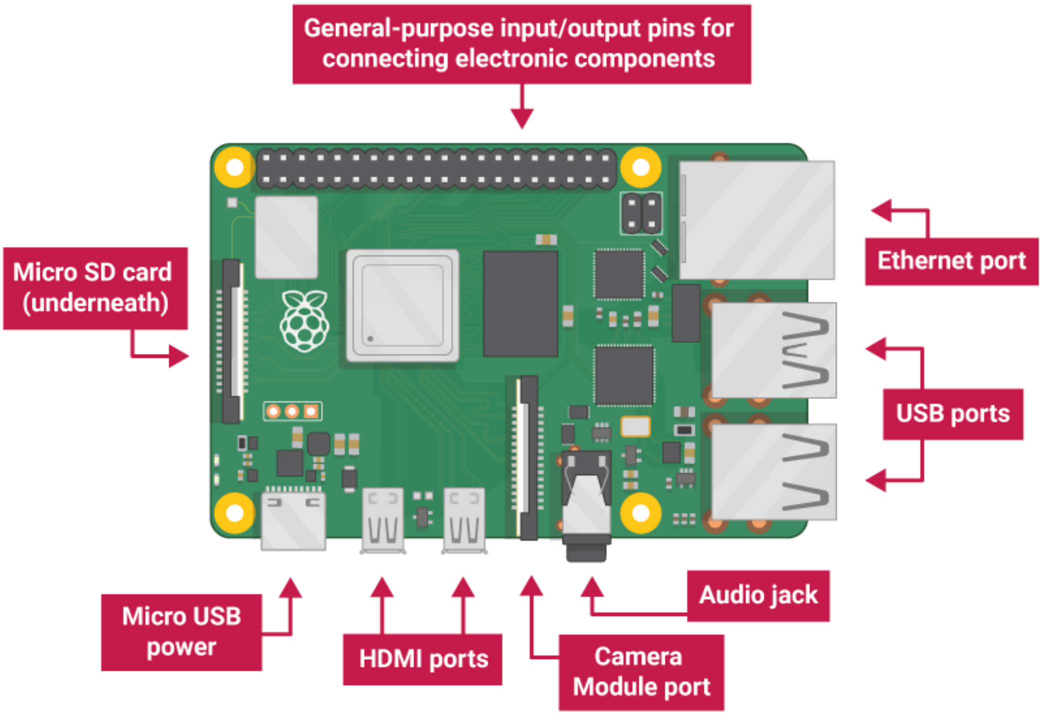

Figure 19: Meet Raspberry Pi

- **USB Ports:**

  I used to connect a mouse and keyboard. I can also connect other components, such as a USB drive.

- **SD Card Slot:**

  I used to slot the SD card in here. This is where the operating system software and my files are stored.

- **Ethernet Port:**

  I used to connect Raspberry Pi to a network with a cable RPI can also connect to a network via wireless LAN.

- **Audio Jack**

I used to connect headphones or speakers here.

- **HDMI port**

  I used connect the monitor (or projector) that you are using to display the output from the RPI. If your monitor has speakers, you can also use them to hear sound.

- **Micro USB Power Connector**

  I used to connect a mouse and keyboard. You can also connect other components, such as a USB drive.

- **GPIO Ports**

  I used to connect electronic components such as LEDs and buttons to RPI

5.2 Setup Raspberry Pi

To get started with Raspberry Pi, I need an operating system. NOOBS is an easy operating system install manager for the Raspberry Pi.

1. **Install NOOBS On SD Card:**
   - **Download**
     a. Using a computer with an SD card reader, visit the Downloads page.
     b. Click on NOOBS, then click on the Download ZIP button under 'NOOBS (offline and network install)', and select a folder to save it to.
     c. Extract the files from the zip.
   - **Format SD card**
     a. Visit the SD Association's website and download SD Formatter 4.0 for either Windows or Mac.
     b. Follow the instructions to install the software.
     c. Insert SD card into the computer or laptop's SD card reader and make a note of the drive letter allocated to it.
     d. In SD Formatter, select the drive letter for SD card and format it.
   - **Drag and drop NOOBS files**
     a. Once SD card has been formatted, drag all the files in the extracted NOOBS folder and drop them onto the SD card drive.
     b. The necessary files will then be transferred to SD card.

c. When this process has finished, safely remove the SD card and insert it into RPI.

2. Connect Raspberry Pi:

- **Fisrt Booting the Raspberry Pi**

a. Plug in keyboard, mouse, and monitor cables.

b. Now plug the USB power cable into RPI.

c. Raspberry Pi will boot, and a window will appear with a list of different operating systems that you can install. We recommend that you use Raspbian – tick the box next to Raspbian and click on Install.

d. Raspbian will then run through its installation process. Note that this can take a while.

e. When the install process has completed, the Raspberry Pi configuration menu (raspi-config) will load.

Figure 20: Connect Raspberry Pi

3. Finish the setup:

(Figure 21: Complete Setup**)**

a. When you start Raspberry Pi for the first time, the Welcome to Raspberry Pi application will pop up and guide you through the initial setup.

b. Click **Next to start** the setup.

c. Set Country, Language, and Time zone, then click **Next** again.

d. **Enter a new password** for your Raspberry Pi and click **Next.**

e. Connect to WiFi network by selecting its name, entering the password, and clicking **Next**.

f. Click **Next** let the wizard check for updates to Raspbian and install them (this might take a little while).

g. Click **Done** or Reboot to finish the setup.

Figure 21: Complete Setup

5.3 Make SSH

I can access the command line of a Raspberry Pi remotely from another computer or device on the same network using SSH.

- **Set up local network and wireless connectivity**

  I make sure Raspberry Pi is properly set up and connected using wireless networking, this can be enabled via the desktop's user interface or using the command line.

1. Display information about the current network status, including the IP address.

Command:

```
ifconfig
```

2. Display the IP addresses associated with the device.

Command:

```
hostname –I
```

- **Enable SSH**

Raspberry Pi OS has the SSH server disabled by default. It can be enabled manually from raspi-config can be used in the terminal.

1. Enter `sudo raspi-config` in a terminal window
2. Select `Interfacing Options`
3. Navigate to and select `SSH`
4. Choose `Yes`
5. Select `Ok`
6. Choose `Finish`

- **Prepare Raspberry Pi for Remote Desktop**

Raspbian generally comes prepackaged with RealVNC for remote connections, the problem is that RealVNC does not work that well in headless mode, primarily. So I prefer to make use of xrdp, which automatically scales to your desktop resolution and generally provides a better experience.

1. In order to install tightvncserver simply run the following command from a terminal window or via Ssh.

   Command:

   ```
   sudo apt install tightvncserver -y
   ```

2. Once that is complete we can then install xrdp

   Command:

```
sudo apt install xrdp -y
```

- **Connect to Raspberry PI using RDP and windows**

  When using windows there is one additional step. The first is open the Remote Desktop Connection tool. You can typically find this by hitting the Windows Key and type “remote desktop connection” which will bring up the Remote Desktop Program.

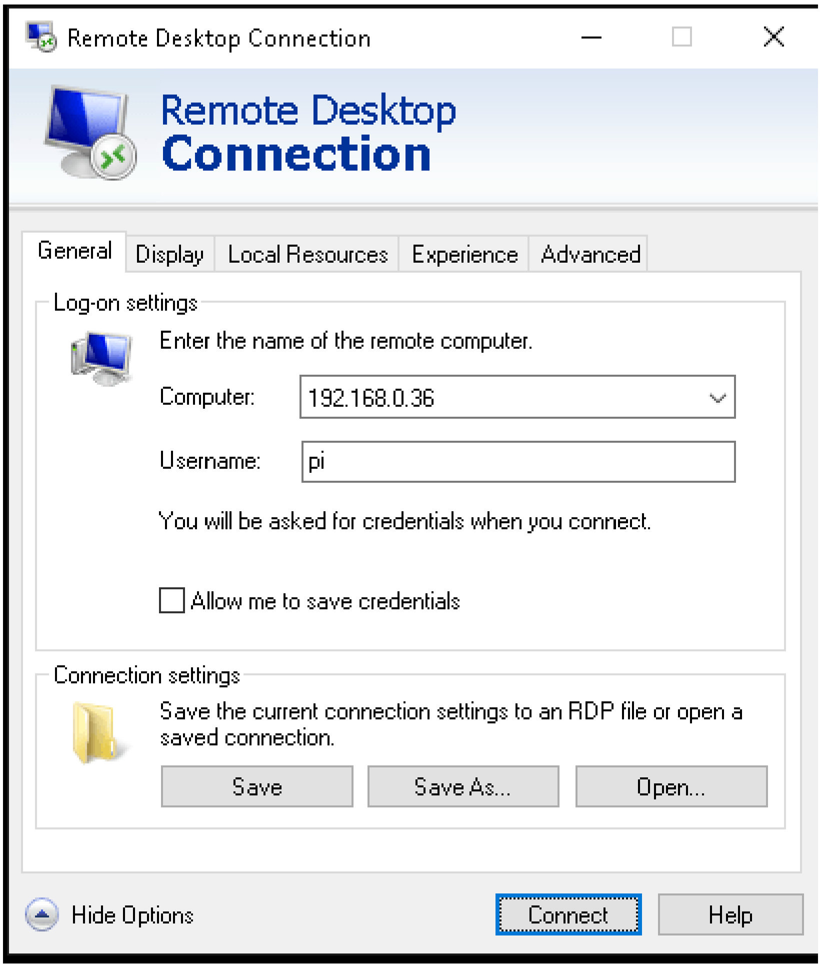

Figure 22: Remote Desktop Connection

5.4 Connect, Configure and Test USB Microphone

Ready to start the Raspberry Pi and insert the mini microphone module as well as audio/headphones.

1. Open the Terminal of the Raspberry Pi.
2. Check the USB connection.

Command:

```
lsusb
```

3. Check the microphone with arecord.

   To list all the playback devices available on Raspberry Pi.

   Command:

```
arecord -l
```

4. Microphone Volume Control.

   To set the volume of playback and recording devices.

   Command:

```
alsamixer
```

5. Records testing.
   I use arecord to record and aplay to play: “-D” to choose the device and “-d” to set the recording time

   Command:

```
arecord -D plughw:1,0 -d 3 test.wav && aplay test.wav
```

5.5 Speech Recognition

- **Requirenment :**

1. Installing Python 2.6, 2.7, or 3.3+

   Before Install anything, I used it to an updated version of packages or their dependencies.

   Command:

```
sudo apt update
```

   a. Install Python3.

      Command

```
sudo apt-get install python3
```

b. Also, I need to install pip.

Command:

```
sudo apt-get install python3-pip
```

c. After the installations are complete, verify that you have pip installed.

Command:

```
pip --version
```

2. PyAudio 0.2.11+

Required because I need to use the microphone input.

Command:

```
sudo apt-get install portaudio19-dev python-all-dev python3-all-dev &&
sudo pip install pyaudio
```

3. Install Google Speech Recognition

Command:

```
sudo pip install SpeechRecognitio
```

4. Create File in the project for use this speech recognition to convert any speech that passes into text.

This is full code in the python file

```
import speech_recognition as sr
r = sr.Recognizer()
mic=sr.Microphone()
while True:
    with mic as source:
        r.adiust_for_ambient_noise(source)
```

```
            audio = r.listen(source)

        try:
              outFile=open(MySpeech.txt , "a")
               outFile.write(" " +r.recognizer_google(audio))
               outFile.write("\n")
        except sr.UnknownValueError:
               print("Error")
```

5.6 Connect Oled Display

- **Pre-requisites** [14]

1. I2C

Ensure that the I2C kernel driver is enabled:

Command:

```
dmesg | grep i2c
```

If you have no kernel modules listed and nothing is showing using dmesg then this implies the kernel I2C driver is not loaded. Enable the I2C as follows:

a. Run **sudo raspi-config**
b. Use the down arrow to select **5 Interfacing Options**
c. Arrow down to **P5 I2C**
d. Select **yes** when it asks you to enable I2C
e. Also, select **yes** when it asks about automatically loading the kernel module.
f. Use the right arrow to select the **<Finish>** button

Next, add your user to the i2c group and install i2c-tools:

Command:

```
sudo usermod -a -G i2c pi
```

```
sudo apt-get install i2c-tools
```

2. SPI

   Enable the SPI port:

   Command:

```
sudo raspi-config
```

   Ensure that the SPI kernel driver is enabled:

   Command:

```
ls -l /dev/spi*
```

   Then add your user to the spi and gpio groups:

   Command:

```
sudo usermod -a -G spi,gpio pi
```

3. Install the latest version of the library Luma.OLED directly from PyPI.

   Command:

```
sudo apt install python-dev python-pip libfreetype6-dev libjpeg-dev build-essential libopenjp2-7 libtiff5
sudo -H pip install --upgrade luma.oled
```

4. Create File in the project for use of this Luma.OLED to show text from output file that converted in speech recognition.

   This is full code in the python file:

```
import time
import os.path
from luma.core.interface.serial import i2c, spi
from luma.oled.device import ssd1306, ssd1309, ssd1325, ssd1331, sh1106
```

5. Run this file

   Command:

```
Python3 OledCode.py
```

Chapter 6

Testing

6.1 Speech Recognition Test

Once a phrase has been detected, the Google Speech Recognition API is used to determine the words that were spoken.

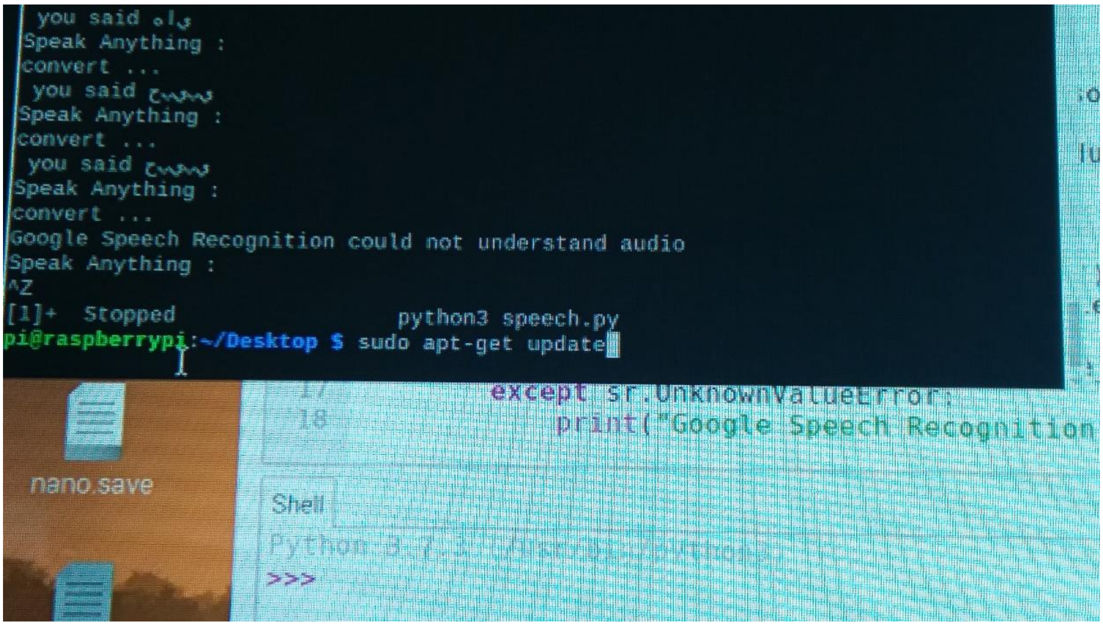

Figure 23: Speech Recognition Test

6.2 Oled Display Test

The display works. I installed the display demo on it.

(Figure 24: Oled Disply Test)

Figure 24: Oled Display Test

6.3 Final Test

I installed the display demo on it.

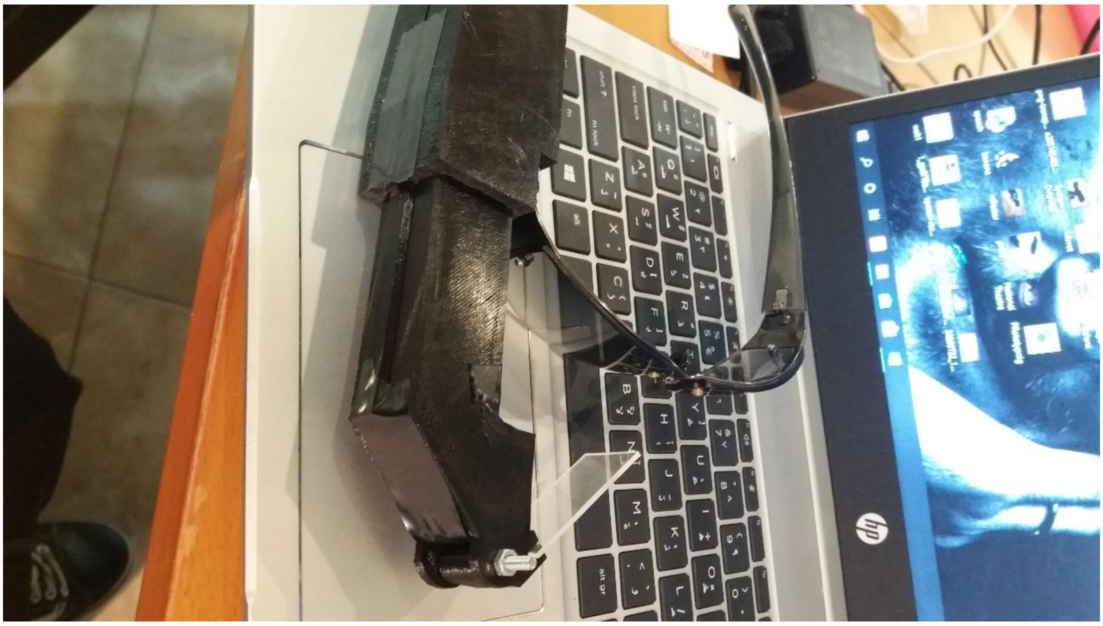

Figure 25: Final Test

Chapter 7
Conclusion

My project was successful, as I was able to achieve my goal: to develop a pair of smart glasses for Deaf or hearing impaired people, which can transcribe the sounds around them, namely the voice of the person they are speaking to, onto the lens of the glasses. Despite this, there are a few improvements to be made to the device.

Some changes I would like to make to the device in the future include changes to increase the functionality of the device. Specifically, I would like to use a more powerful microphone with a higher range so that the device will be able to notify the wearer of speakers from further away than it currently is able to, I would like to support the Arabic language when display on OLED and I would like to make an app on the phone to manage the history data that converted and saved.

A lighter power bank and microphone may be used in order to allow all of the components to be attached to a single platform beside the glasses. The weight of the device can also be reduced by replacing the power bank currently powering the Raspberry Pi with a Lithium Polymer (LiPo) battery, which is smaller and lighter than any power bank.

Lastly, I would like to improve the user interface of the device to allow the wearer of the glasses to change any settings with ease. The device is currently functional; however, a user may find difficulty in operating the device because its user interface has not been finished. The user interface would be designed to accommodate to any needs the wearer of the glasses may have as a result of their hearing disability.

The Reference List

[1] م. ياسين، "الصم والبكم في الوطن العربي.. اهتمام وإهمال،" ساسه، 18 يناير 2018. [متصل]. Available: https://www.sasapost.com/opinion/deaf-and-dumb/.

[2] ا. ا. ا. ل. ا. ل. و. والمنشآت، "واقع الإعاقة في فلسطين،" وكالة الأنباء والمعلومات الفلسطينية- وفا، 2017. [متصل]. Available: http://info.wafa.ps/ar_page.aspx?id=9475.

[3] B. L. Due, "The future of smart glasses:," Centre of Interaction Research and Communication Design, University of Copenhagen, 2014.

[4] OPTISCAN, "WEARABLE TECHNOLOGY - SMART GLASSES," [Online]. Available: https://www.optiscangroup.com/smartglasses.

[5] R. P. Foundation, "Getting started with Raspberry Pi," [Online]. Available: https://projects.raspberrypi.org/en/projects/raspberry-pi-getting-started/2.

[6] G. Moody, "Raspberry Pi OS," 27 September 2013. [Online]. Available: https://en.wikipedia.org/wiki/Raspberry_Pi_OS.

[7] Soren, "Getting started with Python programming and the Raspberry Pi," Raspberry Pi HQ, 6 January 2018. [Online]. Available: https://raspberrypihq.com/getting-started-with-python-programming-and-the-raspberry-pi/.

[8] R. P. FOUNDATION, "Python," [Online]. Available: https://www.raspberrypi.org/documentation/usage/python/.

[9] A. Zhang, "SpeechRecognition 3.8.1," 5 December 2017. [Online]. Available: https://pypi.org/project/SpeechRecognition/.

[10] D. Frants, "LTCCS: The Live-Time Closed Captioning System," 19 July 2017. [Online]. Available: https://www.indiegogo.com/projects/ltccs-the-live-time-closed-captioning-system#/updates/all.

[11] P. Janžekovič, "SMART CAPTION GLASSES, NATIONAL THEATRE, ACCENTURE, EPSON," 27 December 2018. [Online]. Available: https://www.vi-mm.eu/2018/12/27/smart-caption-glasses-national-theatre-accenture-epson/.

[12] M. Breja, "Software Engineering | Prototyping Model," 5 April 2018. [Online]. Available: https://www.geeksforgeeks.org/software-engineering-prototyping-model/.

[13] U. Eriksson, "Why is the difference between functional and Non-functional requirements important?," 5 April 2012. [Online]. Available: https://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/.

[14] https://luma-oled.readthedocs.io/en/latest/install.html, "luma.oled 3.5.0," 5 July 2020. [Online]. Available: https://pypi.org/project/luma.oled/.

[15] G. Moody, "Raspberry Pi," 19 February 2012. [Online]. Available: https://en.wikipedia.org/wiki/Raspberry_Pi.

[16] P. Singh, "Connect, Configure and Test USB Microphone and Speaker with Raspberry Pi," 30 June 2018. [Online]. Available: https://iotbytes.wordpress.com/connect-configure-and-test-usb-microphone-and-speaker-with-raspberry-pi/.

Appendix 1: Information on Appendices

This is Full Code in File Python:

```
import time

import os.path

from luma.core.interface.serial import i2c, spi

from luma.oled.device import ssd1306, ssd1309, ssd1325, ssd1331, sh1106

from luma.core.virtual import viewport

from luma.core.render import canvas

from PIL import Image , ImageFont

import speech_recognition as sr



r = sr.Recognizer()

r.energy_threshold = 3000

r.dynamic_energy_threshold = True



while True:

    with sr.Microphone(chunk_size=1024, sample_rate=16000) as source:

        serial = i2c(port=1, address=0x3C)

        device = ssd1306(serial)
```

```python
        img_path =
os.path.abspath(os.path.join(os.path.dirname(__file__),'images', 'starwars.png'))

        logo = Image.open(img_path)

        virtual = viewport(device, width=device.width, height=768)

        with canvas(virtual) as draw:

            draw.text((20, 20), "Ready ...", fill="red", align="center")



        r.adjust_for_ambient_noise(source ,duration=0.5)

        audio = r.listen(source, timeout=None , phrase_time_limit=5)

        img_path =
os.path.abspath(os.path.join(os.path.dirname(__file__),'images', 'hand.png'))

        logo = Image.open(img_path)

        with canvas(virtual) as draw:

            draw.bitmap((20, 0), logo, fill="white")



        try:

            outF = open("myOutFile.txt", "a")

            outF.write(" "+r.recognize_google(audio ,language= "en-US")+"\n")

            outF.close()

            OutTxt = open("myOutFile.txt" , "r")
```

```
            Last_line = OutTxt.readlines()[-1]

            x = device.width

            with canvas(device) as draw:

                font = ImageFont.truetype('Fonts/FreeSerif', 14)

                w, h = draw.textsize(Last_line, font=font)

            virtual = viewport(device, width=max(device.width, w + x + x),
    height=max(h, device.height))

            with canvas(virtual) as draw:

                draw.text((x, 0), Last_line, fill="white")

            i = 0

            while i < x + w:

                virtual.set_position((i, 0))

                i += speed



        except sr.UnknownValueError:

        print("Google Speech Recognition could not understand audio")
```