

Palestine Polytechnic University
College of IT and Computer Engineering
Computer System Engineering Department



Project title:

Smart Pathfinder for Helping Blind People.

Student Name:

Taima' Aburayyan.

Alaa Baradiyyah.

Israa Abu-arish.

Instructor:

Dr. Radwan Tahboub.

25 AUG 2021.

Table of Content:

Acknowledgment:	5
Abstract :	6
Chapter 1: Introduction	7
1.1: Overview:	7
1.2: Motivation:	7
1.3: Problem Statement:	8
1.3.1: Problem Analysis:	8
1.4: Objectives And Expected Results:	8
1.5: List of Hardware Requirements:	9
1.6: Context Diagram:	9
1.7: Summary :	10
Chapter 2: Background	11
2.1: Theoretical background and literature review:	11
2.1.1: Images Processing:	11
2.1.2: Tesseract OCR:	13
2.1.3: Object detection:	14
2.2: Short description of the parts used in the system:	15
2.3: Specification and design constraints and alternatives:	18
2.4: Summary:	19
Chapter 3: System Design	20
3.1: Brief Description of the System:	20
3.2: Detailed design :	21
3.3: Block diagrams :	22
3.4: System Flowchart:	23
3.5: Summary:	23
Chapter 4 : Software and Hardware Implementation:	24
4.1 Operating System Software:	24
4.2 Software Implementation tool:	24
4.2.1: Node-RED Environment:	24
4.2.2: Tesseract OCR :	25
4.2.3: IBM Watson Services :	26
4.3 Hardware Implementation:	26
4.3.1: Power Source:	26

4.3.2: System hardware :	26
4.3.3: Raspberry PI Camera Interfacing:	27
4.3.4: Ultrasonic Sensors and buzzer Interfacing:	37
4.4: JSON File:	42
4.5: Summary :	47
Chapter 5:Validation and Testing: -----	48
5.1 : image processing test :	48
5.2: System's components Testing:	49
5.2.1: Ultrasonic Sensors & Buzzer Testing:	49
5.2.2: Camera Pi Testing:	50
CHAPTER 6: CONCLUSION -----	53
6.1: Challenges:	53
6.2: Future Work:.....	53
References-----	54

List of Figures:

Figure 1: Context Diagram. -----	10
Figure 2 : Overlapping Fields With Image Processing-----	12
Figure 3 : OCR Process Flow. Source [13].-----	14
Figure 4 : Object detection example. Source [22].-----	15
Figure 5: Raspberry Pi 3 model B+.-----	16
Figure 6: Node-Red platform.-----	17
Figure 7: Ultrasonic Distance Sensor.-----	17
Figure 8 : Buzzer. Source [19] -----	17
Figure 9: Raspberry Pi Camera. Source [19].-----	18
Figure 10 : Visual Recognition is discontinued. Source .-----	18
Figure 11 : Conceptual design.-----	21
Figure 12.a : System block diagram 1.-----	22
Figure 13 : System Flow chart.-----	23
Figure 14 : System flow.-----	25
Figure 15.a : System's Hardware 1.-----	27
Figure 16.a : Camera Interfacing.-----	28
Figure 17 : Inject node.-----	29
Figure 18 : Take-photo node.-----	30

Figure 19 : File-in node.	30
Figure 20 : Watch node.	31
Figure 21 : Object-detector node.	32
Figure 22 : Function node.	32
Figure 23: Image Caption Generator node.	33
Figure 24 : Text to speech service.	34
Figure 25 : Text to speech node.	34
Figure 26 : exec node.	35
Figure 27 : file-in node to read text file.	35
Figure 28 : Switch node 1.	36
Figure 29 : Change node 1.	37
Figure 30: Ultrasonic Sensors And Buzzer.	38
Figure 31 : rpi-srf node.	39
Figure 32 : Switch node 2.	40
Figure 33 : Safe & Not Safe Change Nodes Setting.	41
Figure 34 : Buzzer State Change Nodes Setting.	41
Figure 35 : rpi_gpio out node setting.	42
Figure 36: Image processing testing 1.	48
Figure 37 : Image processing testing 2.	49
Figure 38 : Ultrasonic Sensors & Buzzer Testing.	50
Figure 39: Camera testing 1.	51
Figure 40: Camera testing 2.	51
Figure 41: Camera testing 3.	52

ACKNOWLEDGMENT:

In the name of "Allah", the most beneficent and merciful who gave us strength, knowledge and helped us to make this project. Our heartiest thanks due to our parents for their love and kind support which helped us in the project, there are not enough words to describe how thankful we are to them.

It has been a great opportunity to gain lots of experience in real-time projects, followed by the knowledge of how to actually design and analyze real projects. For that, we want to thank all the people who made it possible for students like us. Special thanks to our university (Palestine Polytechnic University) and all staff for the efforts they did to provide us with all useful information and making the path clear for the students to implement all the education periods in real-time project design and analysis. We would like to express our deepest gratitude to our graduation project supervisor Dr. Radwan Tahboub for his patience and guidance throughout the semester.

At last, we would like to thank all the people who helped, supported, and encouraged us to successfully finish our graduation project.

ABSTRACT:

One of the main problems that blind people face is getting around in and out of their homes. Therefore, we will develop the blind stick to act as a pathfinder that helps the blind in their lives as much as possible by taking a picture of the place where they are by using the pi camera, then by using image processing technology the objects will be detected on that image. At the same time, the system will search for the characters on the street label with the street name. Next, the blind receives movement and warning instructions. Therefore, according to the instructions reached to the blind man, he will move. The distance sensors will detect if there are closely surrounding objects to alert and warn the blind about them. Thus, this system will provide the blind with a safer environment and will give them a sense of independence which will help them lead a normal life.

CHAPTER 1: INTRODUCTION

1.1: OVERVIEW:

Over the years, technology has revolutionized our world and our daily lives. In addition, technology has created amazing tools and resources, bringing useful information to our fingertips.

Modern technology has paved the way for multifunctional devices such as smartwatches and smartphones. Computers are faster, more portable, and more powerful than ever before. With all these revolutions, technology has changed our lives in countless ways, including how we work, live, and play. As well, it has made our lives easier, faster, better, and more fun.

In fact, many things can be accomplished with the help of modern sciences and technologies through which a device or tool is developed that helps people perform their daily tasks in general and helps people with special needs in particular.

1.2: MOTIVATION:

Therefore, with the technological world in which we live, we must take advantage of modern science and technologies to develop systems that can help blind people to make their lives easier. Exploring the role of their agents in improving accessibility and increasing the independence of blind people is an interesting research topic in itself. Therefore, it is important to develop more accessible technology in general.

Blind people face challenges in their daily lives in tasks that are taken for granted if you are sighted. Examples are varied and include finding objects, correctly placing items, and identifying different colors, text, or other visual patterns. These difficulties make several common activities hard to achieve without the help of others. To achieve the inclusion of visually impaired people in a society that fights for equal rights, we must promote their independence not only in that it manifests itself in a home environment but also in the school and work environment.

1.3: PROBLEM STATEMENT:

1.3.1: PROBLEM ANALYSIS:

Given the many problems that blind people suffer from, especially the problems of movement and movement inside and outside the house, and in the street in particular, which is a dangerous place for them because of many objects that can harm them such as cars, bikes, etc. Therefore, we as technology developers should take advantage of modern technology to help them by developing systems that act as their eyes, and in this project, we will provide a simple solution to help them as much as possible in the most frequent places of movement and presence of the blind.

1.4: OBJECTIVES AND EXPECTED RESULTS:

In general, we will benefit from the use of modern technologies and the potential role they can play in promoting a higher degree of autonomy for the blind. To do this we need to achieve the following objectives:

- First of all, our project will work on a small environment which will be our university because of some problems that faced us during the implementation of our project.
- Help the blind to find his way as possible by using the IBM Cloud service text-to-speech which will be used to send sound instructions to help the blind know the direction of the path that should take and how to move.
- That goal will achieve, at the beginning, by use image processing technology that produced by the build-in node called object detector to analyze the objects on the image to help the blind to know what is the surrounding objects or obstacles. In addition, the feedback about the object will converted to speech by using text-to-speech service which provide by IBM cloud. Furthermore, the sound will out by using specific node on the Node-RED platform.
- Search for characters in the image using Tesseract OCR, and determines the location of the blind, then send feedback about location and convert it to a sound controlled by the Node-RED platform.

- According to the returned values from the Ultrasonic Distance Sensors, it will check if there are surrounding objects. And the Magnetic Buzzer will alert the blind person that there are objects around him.

1.5: LIST OF HARDWARE REQUIREMENTS:

To achieve that, we want to use *Raspberry Pi* as the microcontroller which will be programmed by using the *Node-Red platform* and *IBM Cloud*.

Also, we may need to use the following Hardware Requirements:

- Raspberry Pi camera.
- Three of Ultrasonic Distance Sensor (front, right & left of the stick).
- Magnetic Buzzer.
- Raspberry Pi fan.
- Headphones.

1.6: CONTEXT DIAGRAM:

The figure below shows the context diagram of the system:

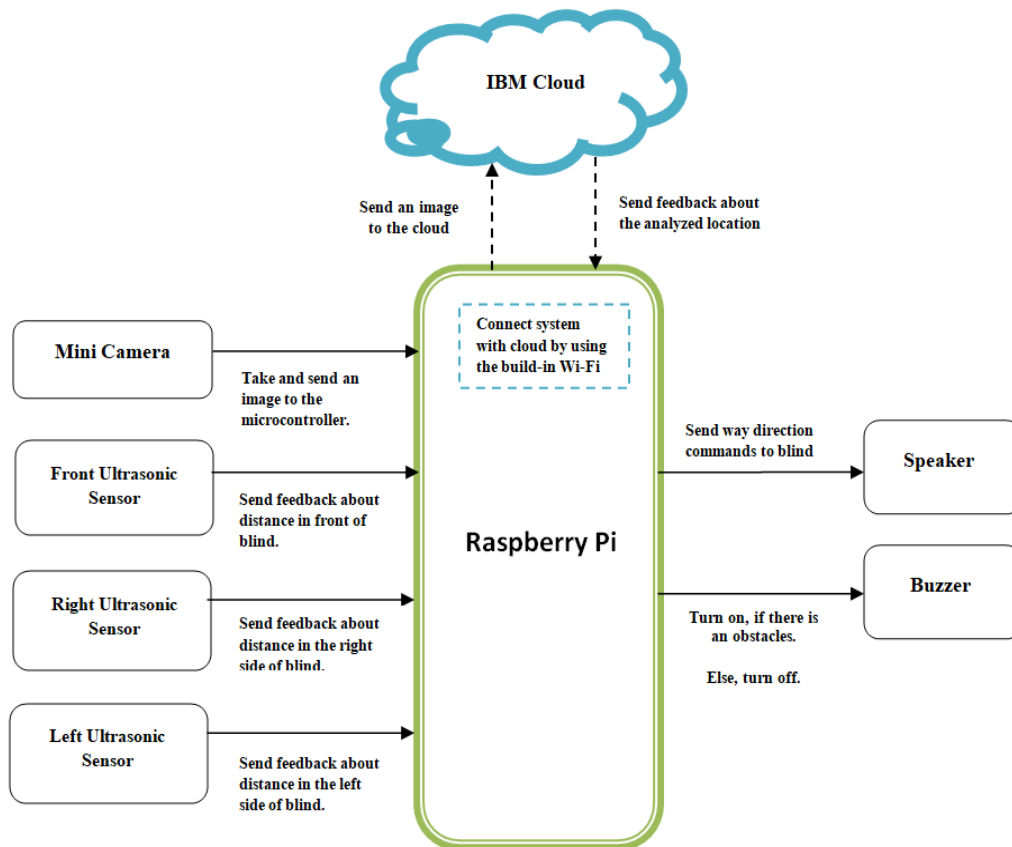


Figure 1: Context Diagram.

1.7: SUMMARY:

In this chapter, we have explained why we chose this idea, our motivation, the problem we want to solve, the goals we want to achieve, and given a simple description of the system. In the next chapter, we will provide the theoretical background and literature review for the techniques we will use in this project.

CHAPTER 2: BACKGROUND

2.1: THEORETICAL BACKGROUND AND LITERATURE REVIEW:

Our project idea has already been implemented before in multiple researches such as PATHFINDER FOR THE BLIND where in this study, they designed a device that help in detecting obstacles about the range of 1.5 meters for the visually impaired persons by providing feedback through vibration and voice communicating the direction that is free. The system hopes to provide a portable unit that can easily be carried and operated by a visually impair user. It could easily be attached to a walking cane [2]. Another study helps to understand the smart navigation system for blind people using raspberry pi in a better way. This smart navigation system is a smart stick that helps the visually impaired or blind people to make their lives simpler and can be seen as an aid that will surpass all the other existing systems [3]. And there is a paper that presents and discusses the results of a multinational survey involving a questionnaire in five languages, as a preliminary investigation of whether blind people might be interested in using a robotic guide and the type of design and features, they would like it to have, this showed that, though there have been a number of projects, only one, Guido, has gone beyond the prototype stage, and this is no longer manufactured [4].

However, those researches were just used to find the right way to cross without using any cloud services that we will use in our project to improve and develop this idea to be more usable and useful.

This project basically includes *Images Processing*, and *Machine Learning*. Therefore, we will give a review for each one in the following sections;

2.1.1: IMAGES PROCESSING:

“Digital image processing is the use of a digital computer to process digital images through an algorithm. As a subcategory or field of digital signal processing, digital image processing has many advantages over analog image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and distortion during processing.” [9].

Simply, *DIP* focuses on developing a computer system that is able to perform processing on an image. The input of that system is a digital image and the system processes that image using efficient algorithms and gives an image or the required feedback as an output.

In *DIP* there are methods and procedures for interpreting digital images for image enhancement and restoration and performing operations on images such as (blurring, zooming, sharpening, edge detection, etc.). *DIP* is a subfield of signals and systems, and it also includes digital electronics, basics of calculus, probability, and differential equations [6].

Since images are defined over two dimensions (perhaps more) digital image processing may be modeled in the form of multidimensional systems. The generation and development of digital image processing are mainly affected by three factors:

1. The development of computers
2. The development of mathematics (especially the creation and improvement of discrete mathematics theory)
3. The demand for a wide range of applications in the environment, agriculture, military, industry, and medical science has increased.

Nowadays, this operation became easier because it has now been offered as a service available on different cloud computing platforms such as Amazon, Microsoft Azure, IBM Cloud, etc. In our project we will use the IBM Cloud services.

2.1.1.1: OVERLAPPING FIELDS WITH IMAGE PROCESSING:

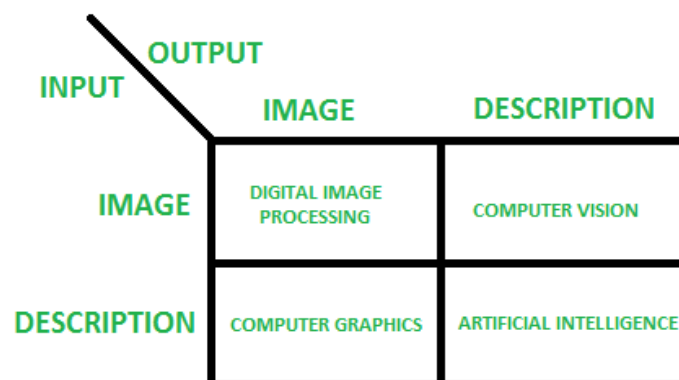


Figure 2 : Overlapping Fields with Image Processing

- ⇒ According to block 1, if input is an image and we get out image as an output, then it is termed as Digital Image Processing.
- ⇒ According to block 2, if input is an image and we get some kind of information or description as an output, then it is termed as Computer Vision.
- ⇒ According to block 3, if input is some description or code and we get image as an output, then it is termed as Computer Graphics.
- ⇒ According to block 4, if input is description or some keywords or some code and we get description or some keywords as an output, then it is termed as Artificial Intelligence.

2.1.2: TESSERACT OCR:

Tesseract is an open-source OCR engine that has gained popularity among OCR developers. It allows to recognize the text in image and supports more than 100 languages. And it is an open-source project, available under the Apache License 2.0. Also, can be used with many programming languages through wrappers or directly from the command line. Tesseract began as a Ph.D. research project in HP Labs, Bristol. It gained popularity and was developed by HP between 1984 and 1994. In 2005 HP released Tesseract as open-source software. Since 2006 it is developed by Google.

Optical Character Recognition (OCR) systems transform a two-dimensional image of text, that could contain machine printed or handwritten text from its image representation into machine-readable text. OCR as a process generally consists of several sub-processes to perform as accurately as possible. The sub-processes are:

- Preprocessing of the Image.
- Text Localization.
- Character Segmentation.
- Character Recognition.
- Post Processing.

The sub-processes in the list above can differ, but these are roughly steps needed to approach automatic character recognition. In OCR software, it's main aim to identify

and capture all the unique words using different languages from written text characters.

Tesseract is an open-source text recognition (OCR) Engine, available under the Apache 2.0 license. It can be used directly, or (for programmers) using an API to extract printed text from images. It supports a wide variety of languages. Tesseract doesn't have a built-in GUI, but there are several available from the 3rdParty page. Tesseract is compatible with many programming languages and frameworks through wrappers. It can be used with the existing layout analysis to recognize text within a large document, or it can be used in conjunction with an external text detector to recognize text from an image of a single text line [20].

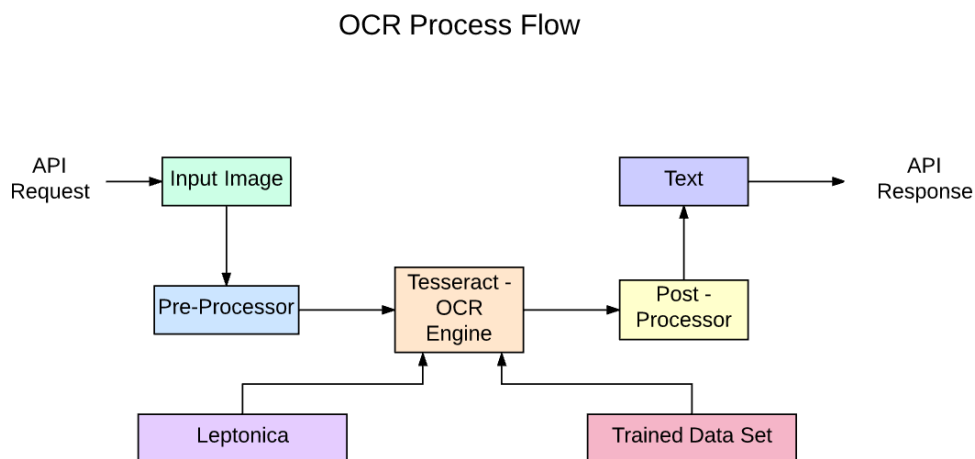


Figure 3 : OCR Process Flow. Source [13].

2.1.3: OBJECT DETECTION:

Object detection is a computer vision technology that allows us to locate objects in an image or video clip. Specifically, object detection draws bounding boxes around these detected objects, allowing us to determine where objects are located (or how they move) in a particular scene. With this type of identification and localization, object detection can be used to count the number of objects in a scene, determine their exact locations, and track them, all while accurately tagging them.

In general, object detection can be divided into machine learning-based approaches and deep learning-based approaches.

In traditional ML-based methods, computer vision techniques are used to look at various features of an image, such as a color histogram or edges, to determine which groups of pixels might belong to an object. These features are then fed into a regression model that predicts the location of the object along with its label. On the other hand, deep learning-based methods use convolutional neural networks (CNNs) to perform comprehensive detection of unsupervised objects, where features need not be identified and extracted separately. In practice, deep learning methods have become the newest method for detecting objects [\[21\]](#) .

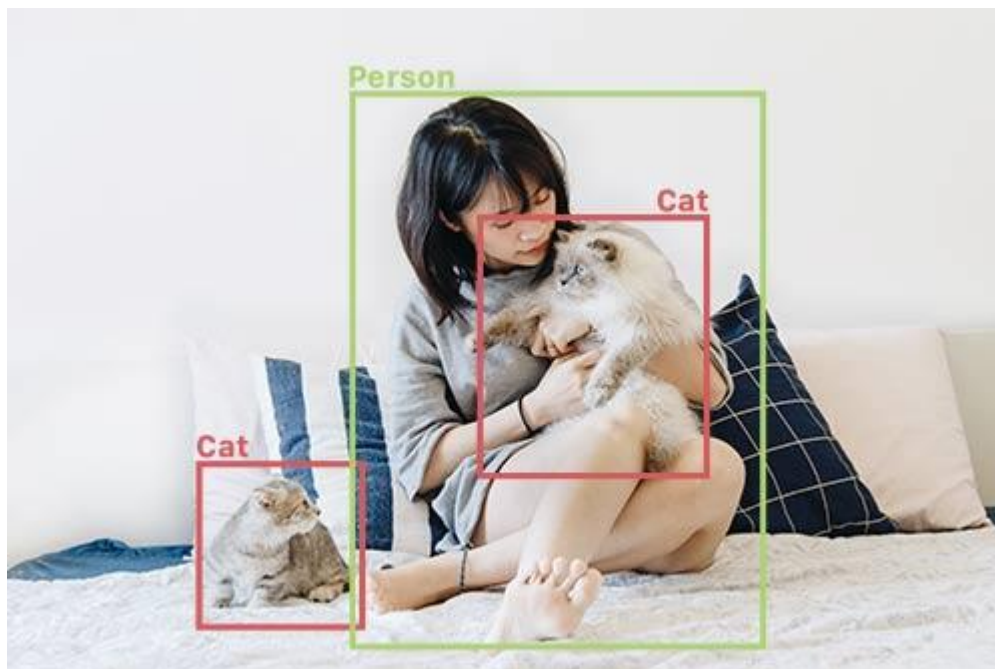


Figure 4 : Object detection example. Source [\[22\]](#).

2.2: SHORT DESCRIPTION OF THE PARTS USED IN THE SYSTEM:

In this project, we need to use the following tools to achieve our objectives:

- **Raspberry Pi:** is a single-board, low-cost, high-performance computer developed in the UK by Raspberry Pi Foundation. It is now widely used in many fields due to its low cost and high portability [\[12\]](#).



Figure 5: Raspberry Pi 3 model B+.
Source [15].

- **IBM Cloud:** it provides a full-stack, public cloud platform with a variety of products in the catalog, including options for computing, storage, networking, end-to-end developer solutions for app development, testing and deployment, security management services, traditional and open-source databases, and cloud-native services [11].
- **IBM Watson services:** IBM's portfolio of enterprise-ready pre-built applications, tools and runtimes are designed to reduce the costs and hurdles of AI adoption while maximizing outcomes and responsible use of AI [1].
- **Node-RED:** Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. Node-RED is battle-tested, open-sourced, and production-ready. It was originally created by the IBM Emerging Technology organization. It is included in IBM's Bluemix (a Platform-as-a-Service or PaaS) IoT starter application package. Node-RED can also be deployed separately using the Node.js application. At present, Node-RED is a JS Foundation project. It provides a web browser-based flow editor, which we will use to control Raspberry Pi with the peripherals [10].
The figure below shows an example of how the Node-RED work:

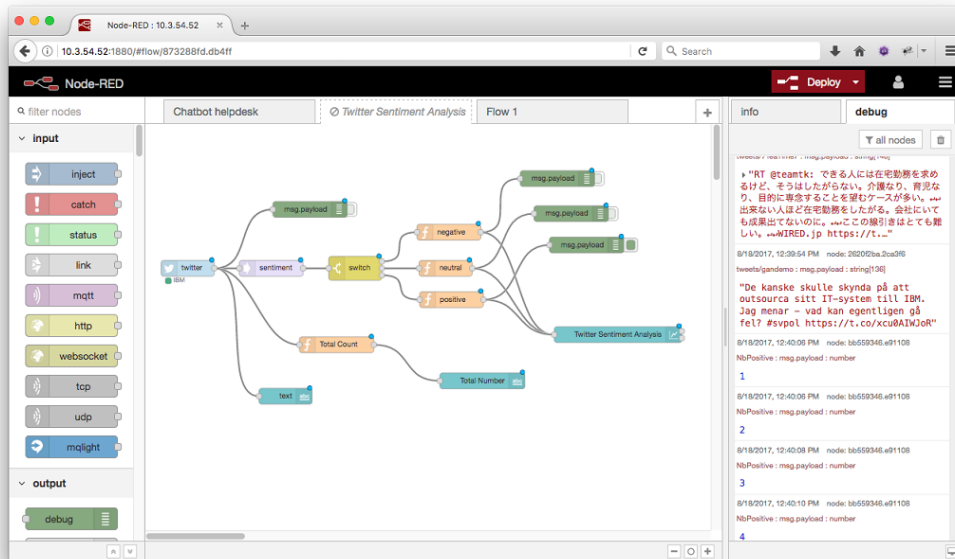


Figure 6: Node-Red platform.
Source [16].

- **Ultrasonic Distance Sensor:** an ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal.



Figure 7: Ultrasonic Distance Sensor.
Source [17].

- **Magnetic Buzzer:** turn on when there is an obstacle.



Figure 8 : Buzzer. Source [19]

- **Raspberry Pi Camera:** to take a picture and send it to IBM cloud, so the robot can know the way that must be taken, for guiding the blind people.

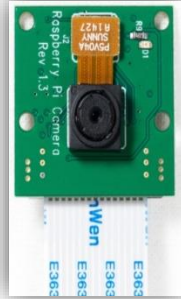


Figure 9: Raspberry Pi Camera. Source [19].

2.3: SPECIFICATION AND DESIGN CONSTRAINTS AND ALTERNATIVES:

In this project, we faced some challenges and constraints which are as follow:

- ◆ During the implementation of this project, we faced a problem with the most important tool, the IBM Watson visual recognition service that was supposed to be used to train the system on the usual paths for the blind. Unfortunately, this service has been disabled this year (see Figure H).

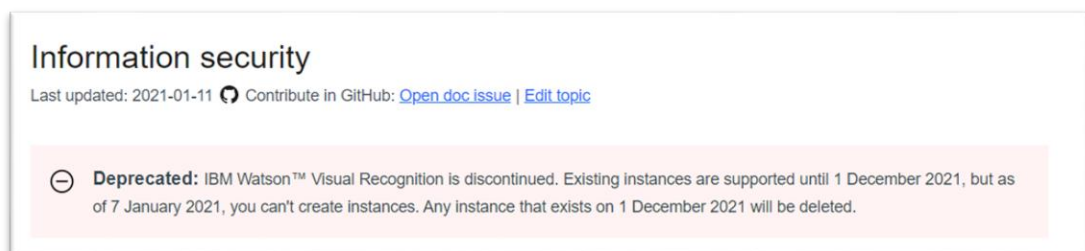


Figure 10 : Visual Recognition is discontinued. Source.

- ◆ Firstly, to solve this problem, we looked for other platforms that support similar services that we can use in the Node-RED platform and found AWS Visual Rekognition and Google Image Recognition Vision AI but they are not free.

- ◆ In addition, we found that TensorFlow is supported in the Node-RED platform and we tested it but the pre-trained node model takes about 3 minutes to give a result where the blind need less time to get the feedback from the system.
- ◆ Furthermore, we tried using OpenCV and programming our own model but had problems installing its packages. Also, we asked an expert and he told us, to build our own model, we need a lot of time that we don't really have at that time no matter how much extra time we need to fix the installation issues.
- ◆ In conclusion, we decided to reduce the services of our project from serving the blind in any paths to serve them only in the university community which will be achieved by defining the university street through poster boards containing the street names of the university buildings such as building B, building B + street, and building street C. The system then recognizes the text or characters in the image using the open-source text recognition engine Tesseract OCR to tell the blind where he/she is.

2.4: SUMMARY:

Previously in this chapter, we provided the theoretical background and literature review of the techniques we will use to implement the system such as image processing, optical character recognition provided by Tesseract engine, IBM Cloud Watson, object detection technology, and etc. Also, describe the hardware tools we will use. In addition, we explained the challenges and limitations encountered during the implementation of the project. In the next chapter, we will explain the system design in detail.

CHAPTER 3: SYSTEM DESIGN

In this chapter, we will determine our project design in detail. First of all, this system based on cloud services (i.e., IBM cloud & IBM Watson) which will be programmed with the Node-Red programming tool and the Tesseract OCR open-source text recognition engine. The system will help the blind person to know the right path as possible and notify him/her if there is any obstacle in his/her path.

However, this chapter describes the conceptual design, detailed design, block diagram, and system flowchart.

3.1: BRIEF DESCRIPTION OF THE SYSTEM:

In this project, we will develop a blind stick work as a pathfinder which will help blind people to determine their right way by taking a shoot of the place or street using the pi-camera, then by using image processing exactly Tesseract OCR open-source text recognition engine that will access the picture file on raspberry pi to scan the image for a character that led to in which street the blind is. After that, by make use of the integration between IBM Cloud Watson services -exactly text-to-speech service- and Node-Red on raspberry, the blind will get the movement instructions and what the objects around him/her as a sound (by headphone). So, according to the command that arrived at the blind will move. And by using the three Ultrasonic Distance Sensors that will be placed on the front, right and left of the stick they will detect if there are closely surrounding objects. And with the addition of the Buzzer, will achieve the ability to indicate object detecting procedure to alert and warn the blind person that there are objects around him. Note that, in this system we need to add a street label boards that contain the names of the University buildings streets as Building B Street, Building B+ Street, and Building C Street. Thus, when the pi camera takes a photo of the street, the system will scan for a character in the photo for the purpose to find the street name.

3.2: DETAILED DESIGN:

The Figure below shows the conceptual design of the system where the blind stick will work as a pathfinder that will help blind people in their life as possible, by using cloud service (i.e., IBM Cloud) with the Node-Red platform, Optical Character Recognition, and Watson service, embedded system techniques (i.e., Raspberry Pi 3 Model B+, Raspberry Pi camera, ultrasonic sensor & buzzer) and etc. That will be achieved by taking a shot of the place where they are using the pi-camera, then by using image processing exactly the built-in object detection model on the Node-RED that will analyze the picture to detect the object in it while using the OCR will help the blind person to know if there are any labels on the street hold the name of the it. After that, by making use of the integration between IBM Cloud Watson services and Node-Red on Raspberry Pi, the blind person will get the movement instructions by using the text-to-speech Watson service. Therefore, according to the command that arrived at the blind person will move. And by using the three Ultrasonic Distance Sensors that will be placed on the front, right and left of the stick they will detect if there are closely surrounding objects. And with the addition of the Buzzer, will achieve the ability to indicate object detecting procedure to alert and warn the blind person that there are objects around him.

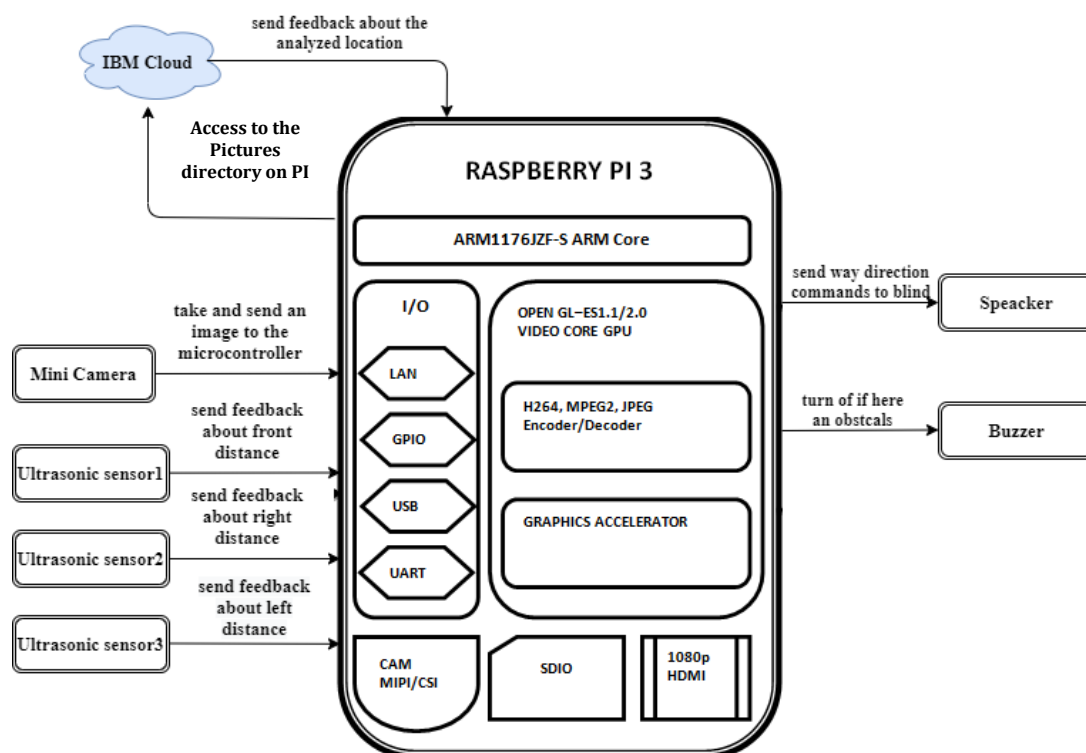


Figure 11 : Conceptual design.

3.3: BLOCK DIAGRAMS:

Figures 11.a & 11.b shows the block diagram of the system:

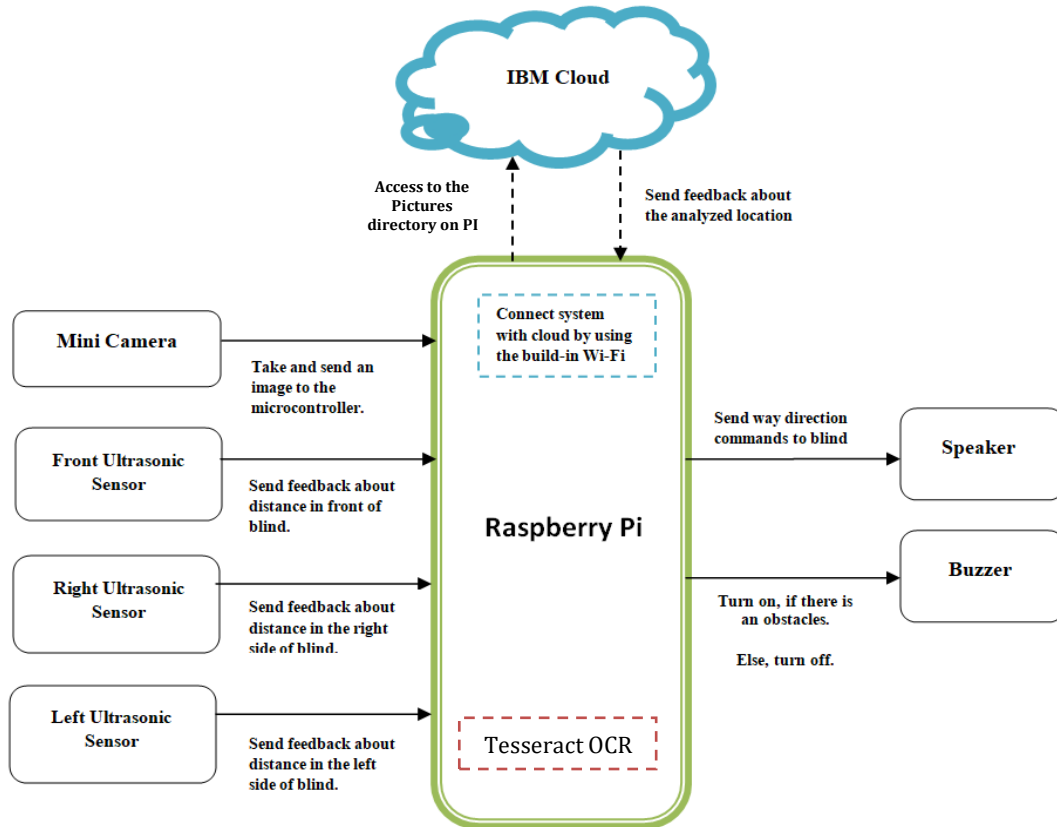


Figure 12. a: System block diagram 1.

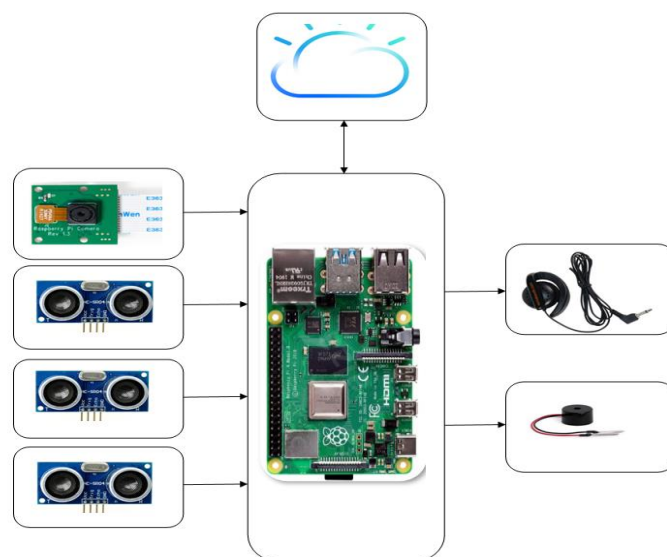


Figure 12.a: System block diagram 1.

3.4: SYSTEM FLOWCHART:

The Figure below shows the System Flowchart:

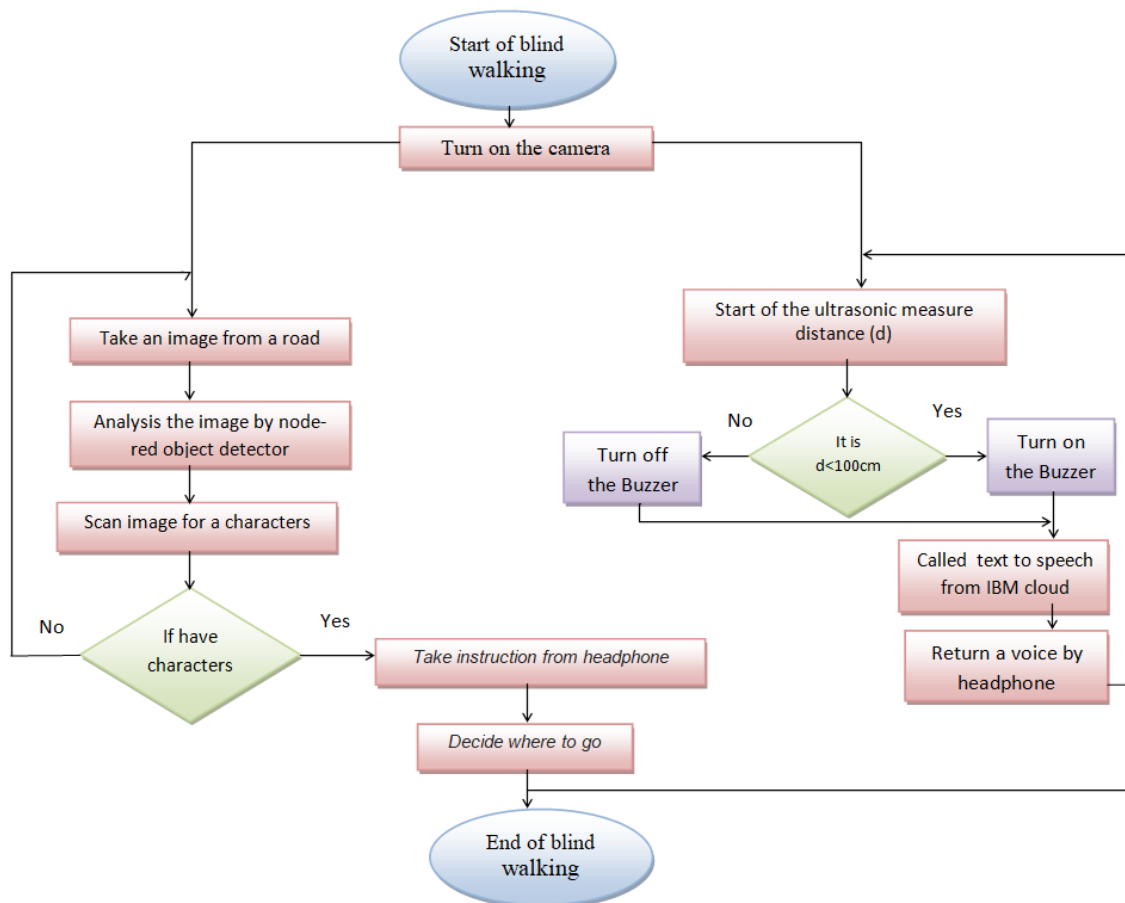


Figure 13 : System Flow chart.

3.5: SUMMARY:

In this chapter, we have given a brief description of the stick system, described the conceptual design, detailed design, block diagram, and system flowchart. In the Next chapter, we will explain system software and hardware implementation, and describe how we programming the system using Node-RED platform.

CHAPTER 4: SOFTWARE AND HARDWARE IMPLEMENTATION:

This chapter describes the software that we used to build our project, such as the operating system, programming platform, and the set of other tools and packages that help us to complete the project.

4.1 OPERATING SYSTEM SOFTWARE:

One Operating Systems were used in our project, which is for the Raspberry PI:

- Raspberry Pi OS (formerly known as Raspbian) 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 Raspberry Pi family of compact single-board computers.

4.2 SOFTWARE IMPLEMENTATION TOOL:

4.2.1: NODE-RED ENVIRONMENT:

Node-RED Platform enables us to stitch together Web services and hardware by replacing common low-level coding tasks, and this can be done with a visual drag-drop interface. Various components in Node-RED are connected together to create a flow. Most of the code needed is created automatically.

The figure below, shows our Node-RED flow that controlled the system:

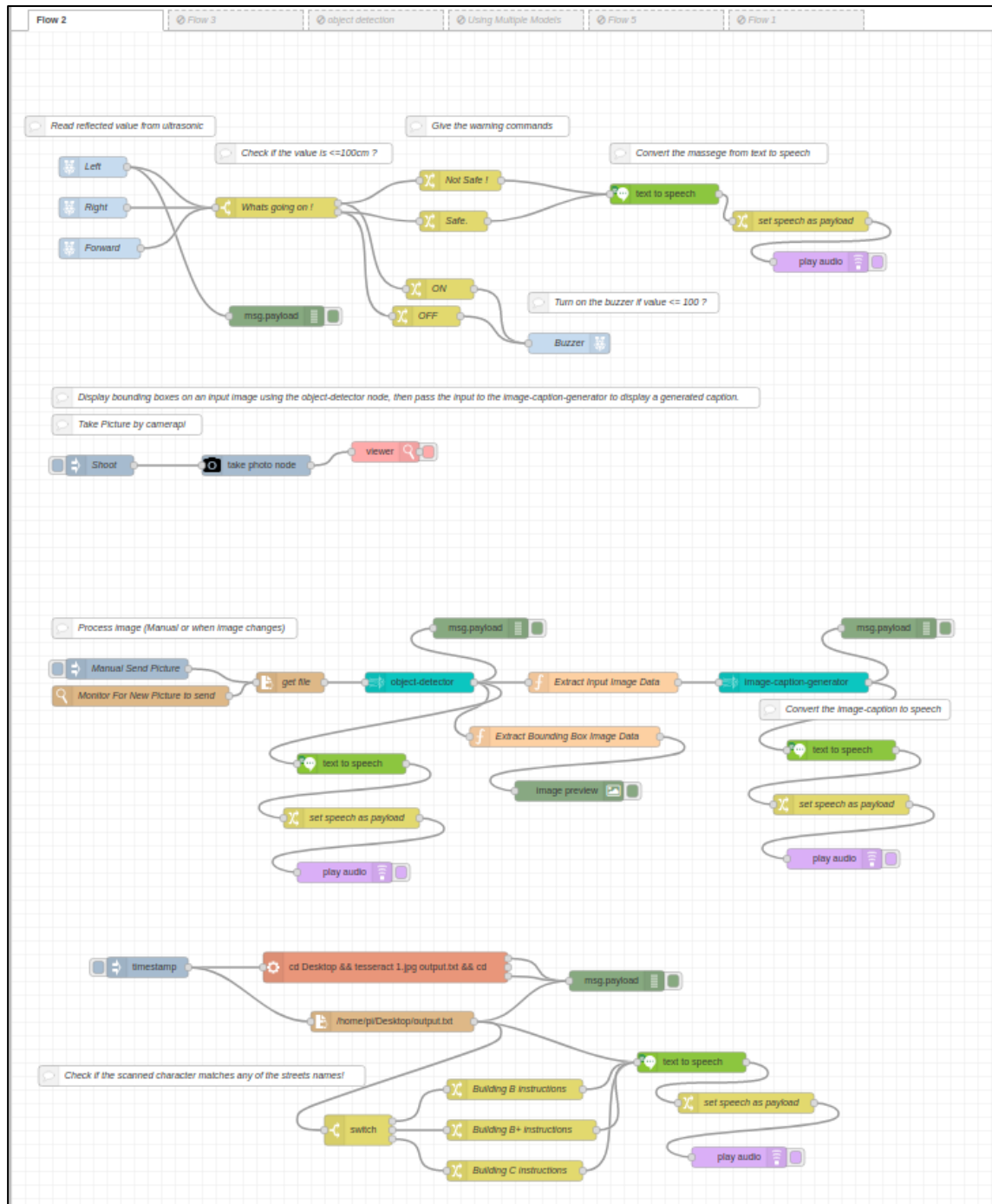


Figure 14 : System flow.

4.2.2: TESSERACT OCR:

We used it directly from the command line by using *exec node* that allows us to take any existing system command, or script that we have written and run it from node-red and incorporate the results in our flow. Thus, we use it to run `<tesseract _name of image_ _name of text file to store the result_>` which will be as `<tesseract`

image.jpg output >. So, the result will be stored in output.txt file that the system flow must access to read it.

4.2.3: IBM WATSON SERVICES:

To achieve our goals, we had created one IBM Watson Services which is text to speech service.

IBM Watson Text to Speech is an API cloud service that enables you to convert written text into natural-sounding audio in a variety of languages and voices within an existing application or within Watson Assistant.

Where, the text to speech service understands the text and natural language to generate synthesized audio output complete with appropriate cadence and intonation. Thus, this service helped us to convert the text (i.e., warning commands) to speech.

4.3 HARDWARE IMPLEMENTATION:

4.3.1: POWER SOURCE:

The Raspberry PI, the ultrasonic sensors, buzzer, and twice fans are powered by a 5v while the pi-camera powered by +3.3v. Thus, the current that the Raspberry PI requires is dependent on what is connected to it, where each ultrasonic sensor requires 15mA to start up, and the buzzer requires 30mA. Therefore, Power Bank is used to provide the system with a 5v and 3A.

4.3.2: SYSTEM HARDWARE:

Our system looks like shown in the figures below:

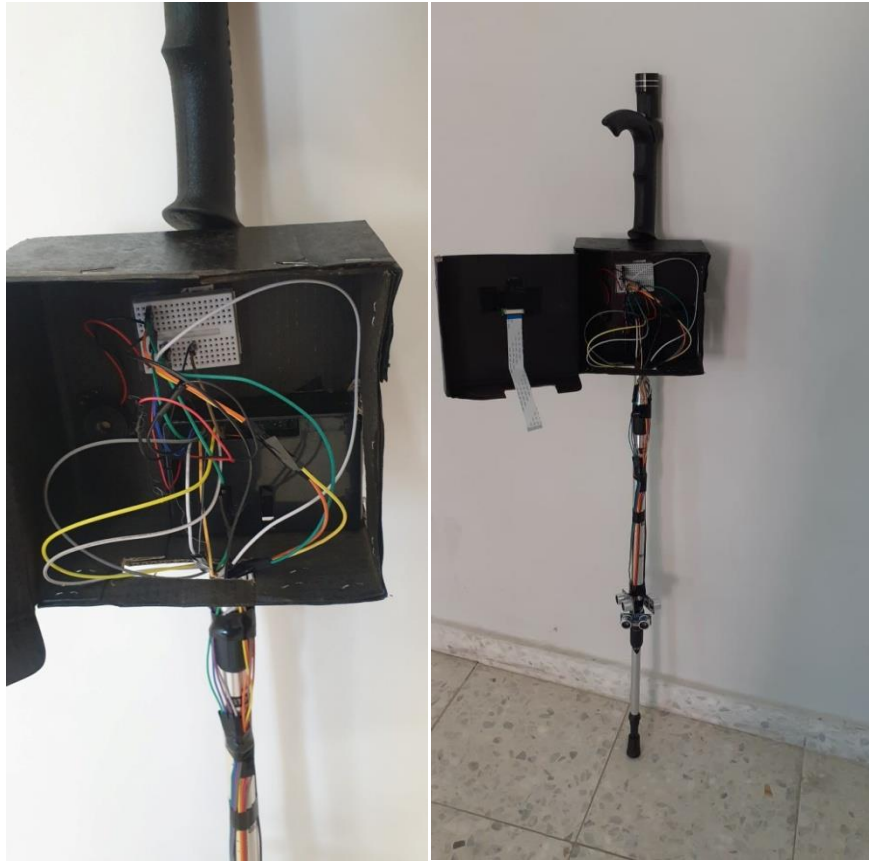


Figure 15.a: System's Hardware 1.



Figure 15.b: System's Hardware 2.

4.3.3: RASPBERRY PI CAMERA INTERFACING:

In this system, we mainly use the camera module to take a picture of the place or path where the blind person is - and save it to /home/pi/Pictures - to help him/her see if there are any objects around and know in which street of university, he is in. This is done by taking advantage of the integration between Raspberry PI and the IBM cloud

and using Tesseract OCR engine. The process flow is shown in the Figures [16.a](#) & [16.b](#) :

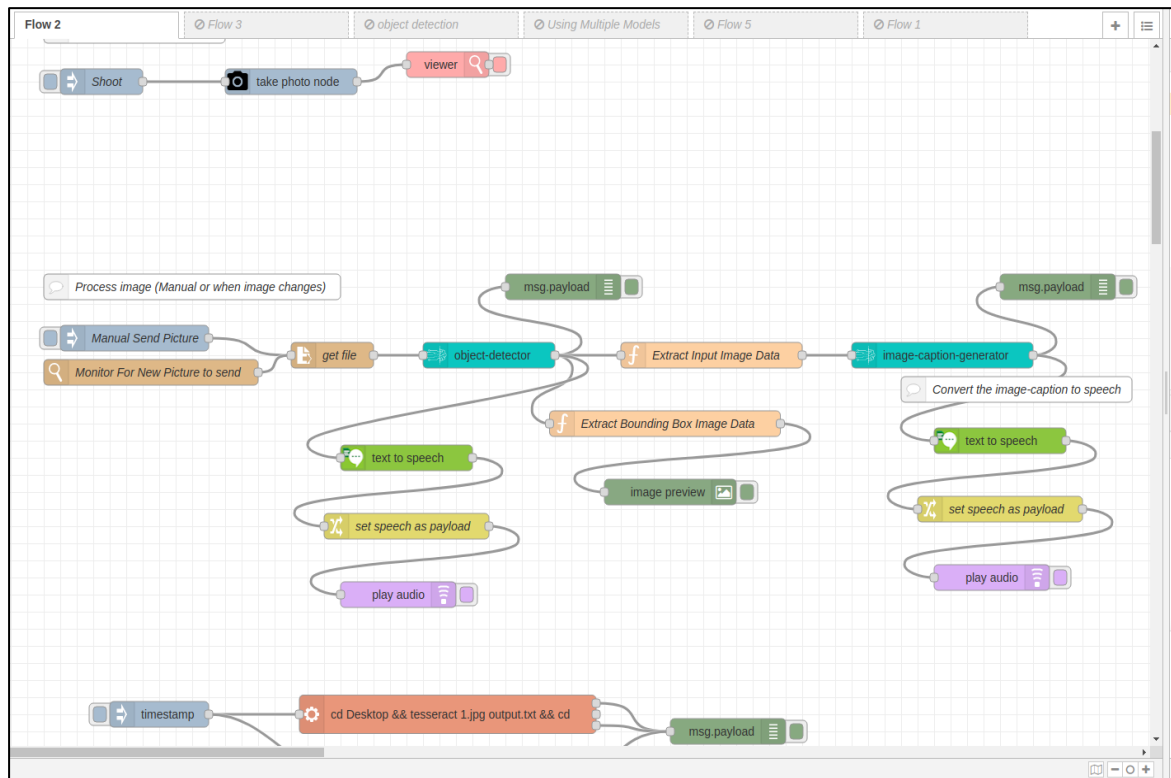


Figure 16. a: Camera Interfacing.

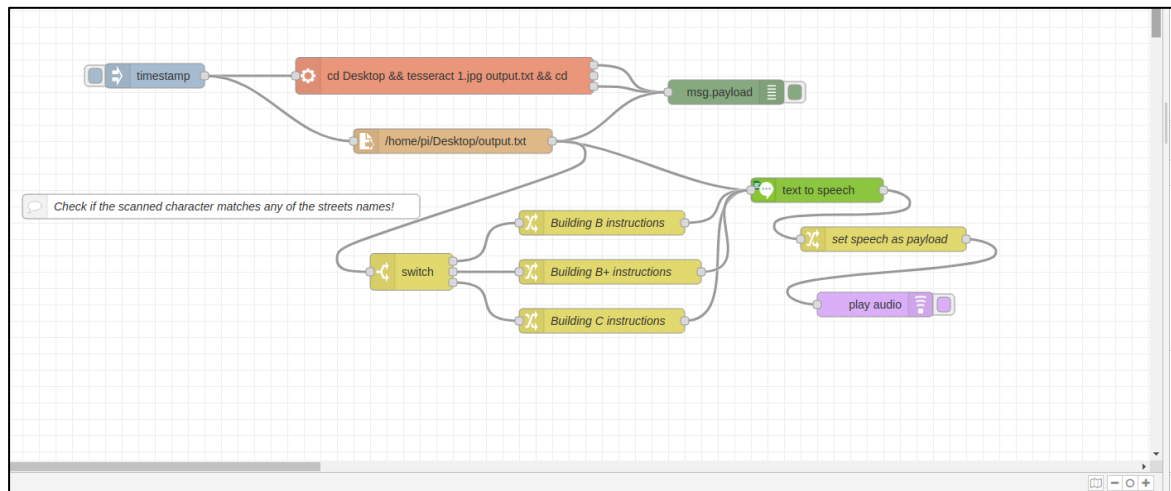
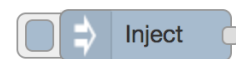


Figure 16.b: Camera Interfacing Count.

- ◆ Inject node: Injects a timestamp or user-configured text into a message. Can be configured to inject manually, at a set interval, or at



specific times. We use it as a timestamp to trigger the camera node at a set interval to be repeated every 10sec. see figure [17]

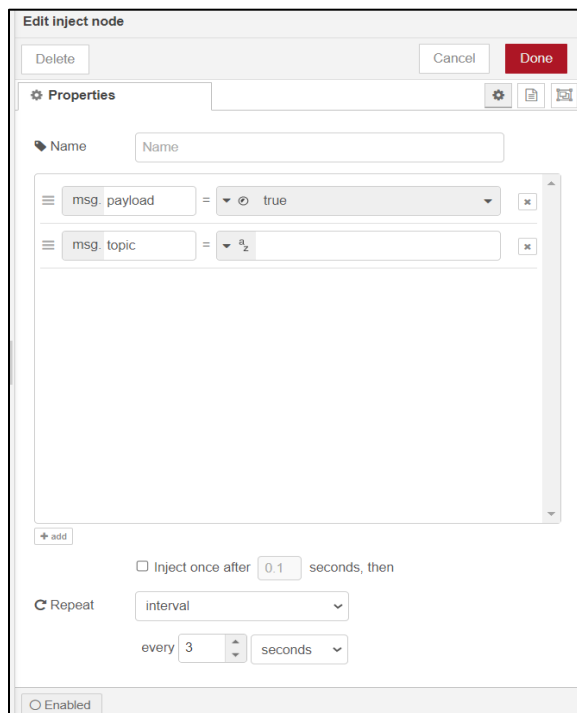
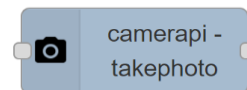


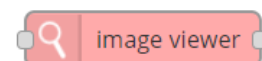
Figure 17 : Inject node.

- ◆ Take-photo node: used to take pictures on Raspberry Pi.



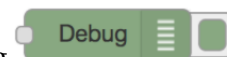
This node will work by enabling the Raspberry Pi Camera. We did the settings as shown in Figure [18] :

- ◆ Image viewer node: its view images in the node-red editor



(For preview / debug purposes). Features include ability to display a jimp image, buffer, file name, base64 string, Data URL, Image URL. So, we used it to help us as programmers to view the image captured by the pi camera.

- ◆ Debug node: it can be used to display messages in the Debug



sidebar within the editor. The sidebar provides a structured view of the messages it is sent, making it easier to explore the message. Also, we used it to help us as programmers.

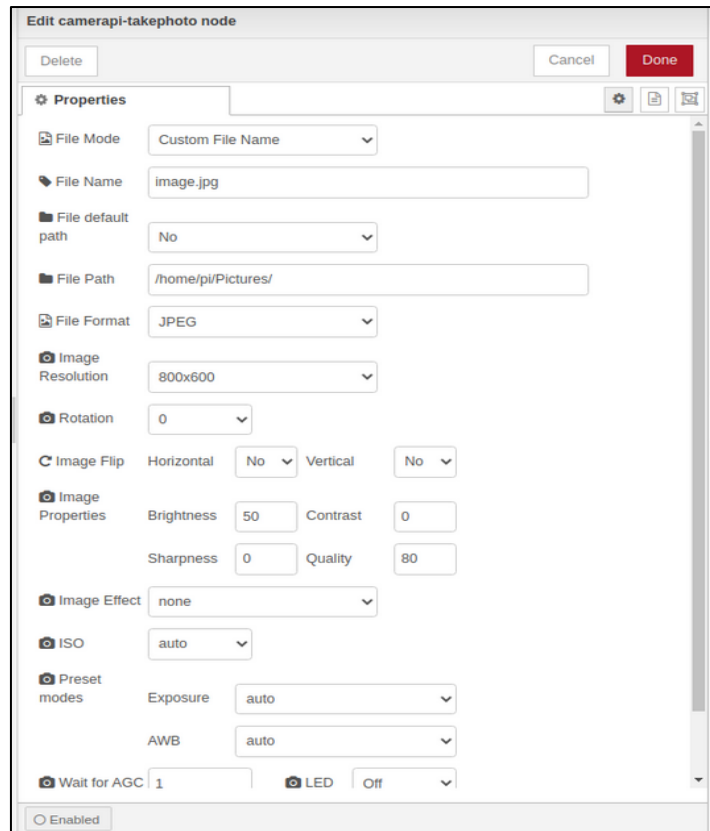


Figure 18 : Take-photo node.

- ◆ File-in node: reads a specified file –the images directory in /home/pi/Pictures- and sends the content as msg.payload, and the filename as msg.filename (see figure [19]).

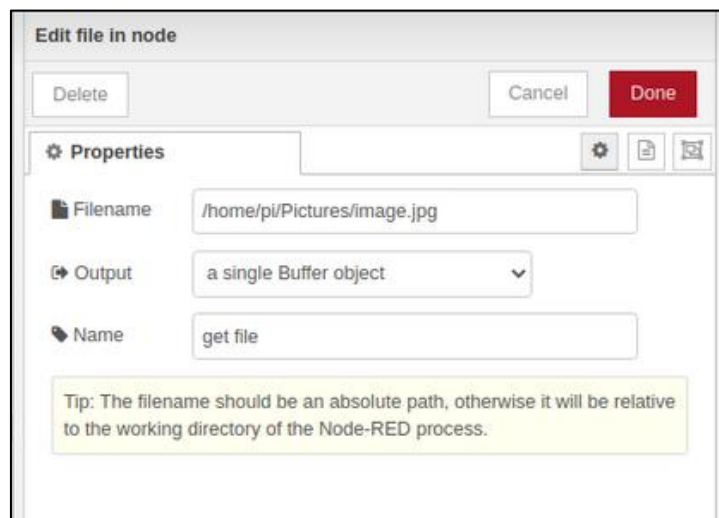
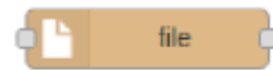
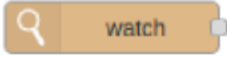


Figure 19 : File-in node.

- ◆ Watch node: Watches the images directory/file for  changes. The full filename of the directory/file that actually changed is put into msg.payload, while a Stringfield version of the watch list is returned in msg.topic (see figure [20]).

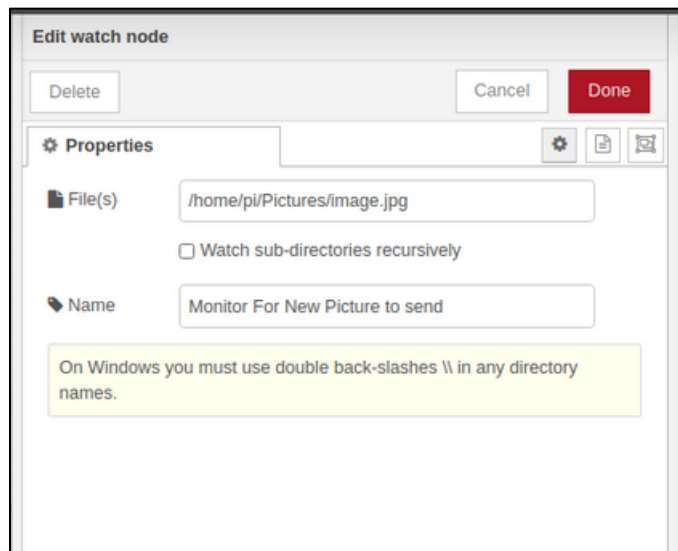
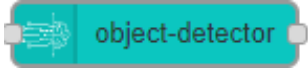


Figure 20 : Watch node.

- ◆ Object-detector node: one of the Node-RED nodes for  deep learning micro-services from the Model Asset eXchange, providing support for common audio, image, video, and text processing tasks. This node used to localize and identify multiple objects in a single image (see Figure [21]). Thus, using this node will help the blind person to know the things around him.

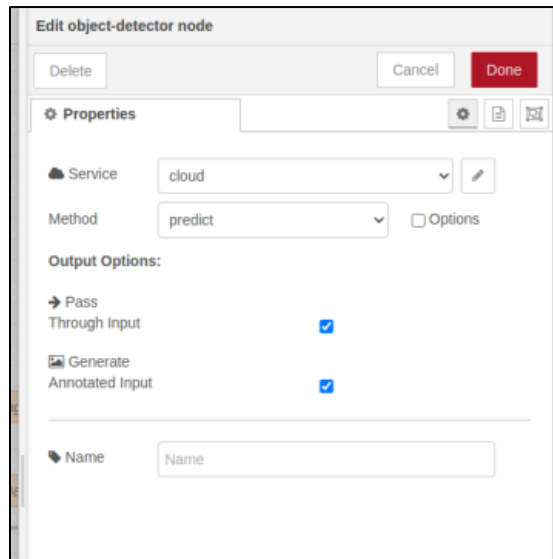
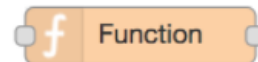


Figure 21 : Object-detector node.

- ◆ Function node: programmable generic node.



With standard JavaScript, a node can be designed to perform complex processing on its input messages, and generate one or more output messages. We used this node to check if the ultrasonic sensor returned a value less than or equal to 100cm to send warning commands to the blind and turn on the buzzer. As long as we are using three sensors, we will need to use three function nodes for each sensor. Thus, in this node we set up its properties by writing the JavaScript code in its On Message properties. Here, this node used to convert the image from message payload format to annotated Input format (see [Figure \[22\]](#)).

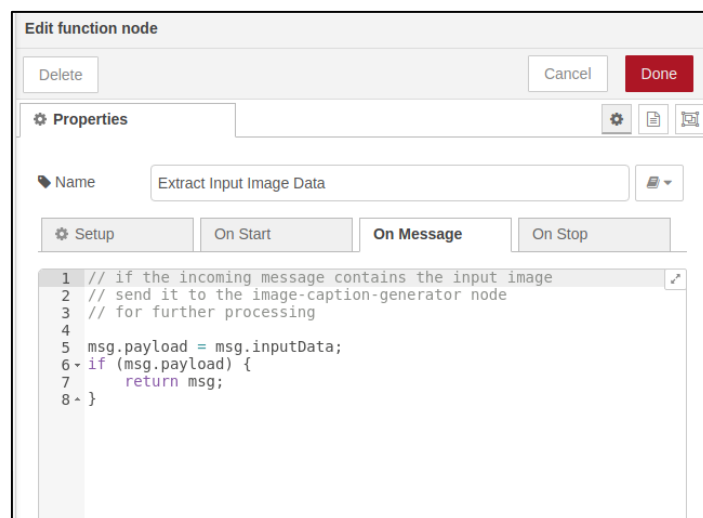
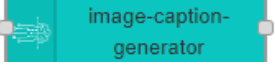


Figure 22 : Function node.

- ◆ Image Caption Generator: one of the Node-RED nodes  for deep learning micro-services from the Model Asset eXchange. This node generates captions that describe the contents of images (see Figure [23]).

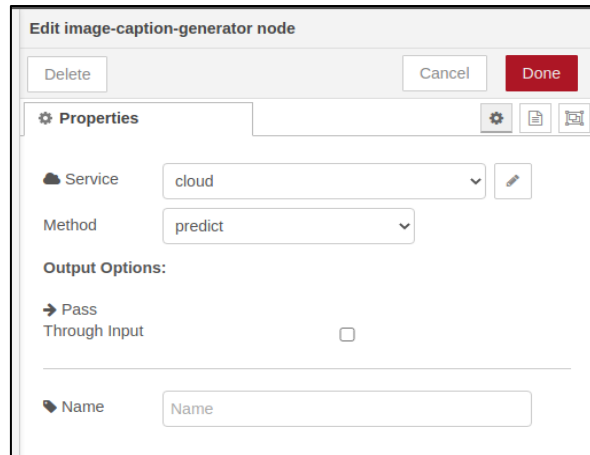
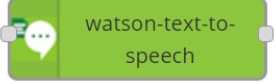
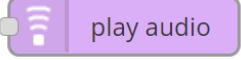


Figure 23: Image Caption Generator node.

- ◆ Watson text-to-speech node: used to Convert  msg.payload that came from the visual recognition node - which contains the information about the analyzed image - into audio speech to tell the blind person about things around him. The properties setting is shown in the Figure [25] where Username, Password, the API key, and Service Endpoint are set up based on text to speech IBM Watson service *credentials* which shown in the Figure [24] .

- ◆ Play audio node: node to play audio from a raw audio  buffer. Works well together with the Watson Text to Speech node, using the WAV audio format. Therefore, it used to play audio that came from the text to speech node that contain the warning commands.

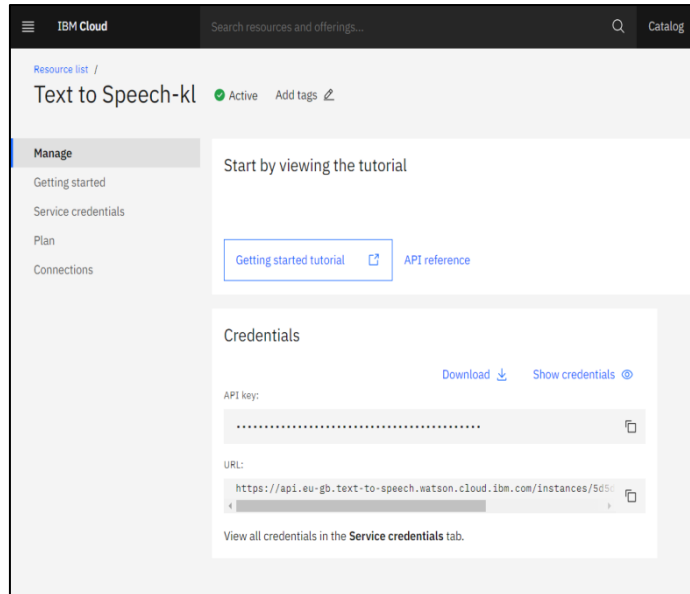


Figure 24 : Text to speech service.

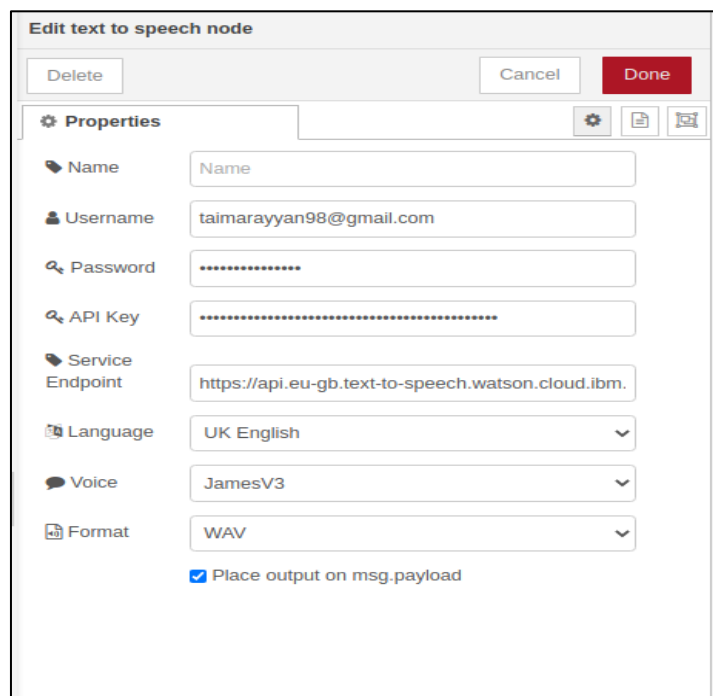
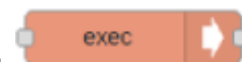


Figure 25 : Text to speech node.

- ◆ Exec node: Calls out to a system command and provides three outputs: stdout, stderr, and return code. By default, uses exec (), which



calls the command, blocks while waiting for completion, and then returns the complete result in one go, along with any errors. This node allows us to take any existing system command, or script that we have written and run it from node-red and incorporate the results in our flow. In this case, it contains this command "cd Picture && tesseract image.jpg output && cd" which used to access Picture folder after that by using tesseract command with the name of the image to scan it for any characters and if find something the result will be stored in output text file (see Figure [26]).

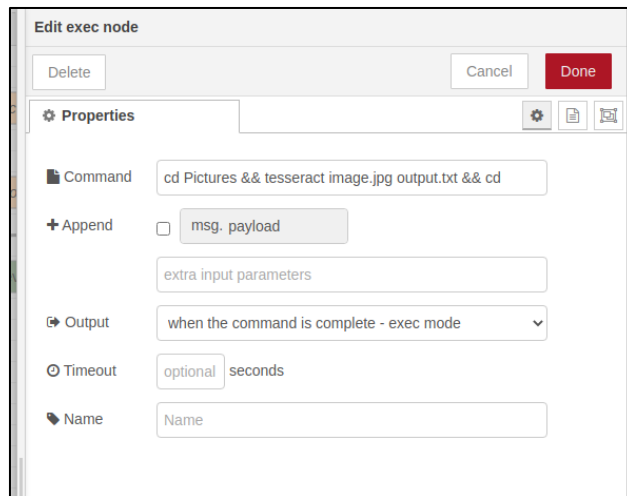


Figure 26 : exec node.

- ◆ In Figure 15.b, we used another file-in node to read the result of "tesseract image.jpg output" command that will store in *output.txt* file (see Figure [27]).

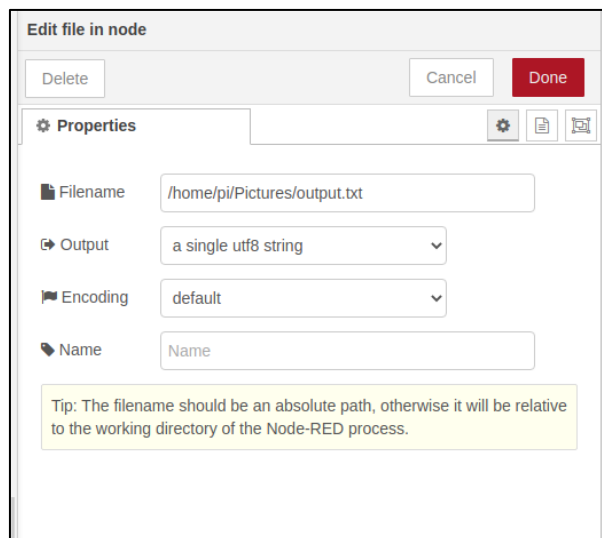
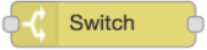


Figure 27 : file-in node to read text file.

- ◆ Switch node: This node routes messages based on them  properties. Properties are configured using the UI and can be a variety of logic (>, <, >= etc.) applied to a message property. In Figure 15.b, we used this node to check if the text file content match one of the street labels as shown in Figure 28], Then the system will output a movement instruction that will tell the blind how many steps he needs to walk to reach the square of the building.

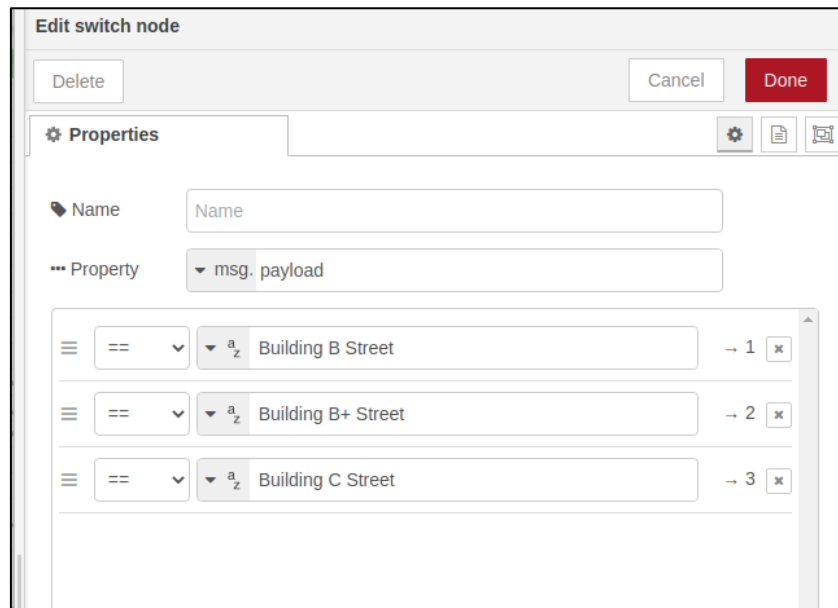
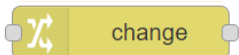


Figure 28 : Switch node 1.

- ◆ Change node: can be used to set, change or delete  properties of incoming messages. A variety of configurable rules allow complex changes including search and replace in the msg.payload. We have used this node to set the msg.payload to string as shown in Figure 29]. Where, it contains the movement instructions that will be sent to the text-to-speech node. The movement instructions will be as follows:
 - Building C instruction: "You are now on Building C Street. If you want to get to Building C square you have to walk about 150 steps, after that turn left, then you have to go up the seven stairs to be inside".
 - Building B+ instruction: " You are now on Building B+ Street. If you want to get to Building B+ square you have to turn left and walk about

10 steps. To be inside the building, turn right and walk about 120 steps straight ahead then you have to go up the stairs. "

- Building B instruction: " You are now on Building B Street. If you want to get to Building B square you have to walk about 200 steps straight ahead. Then, you will find its gate on your left. And in the same path, you will find the cafeteria, to enter just shorten your steps to 8 steps less to be inside ".

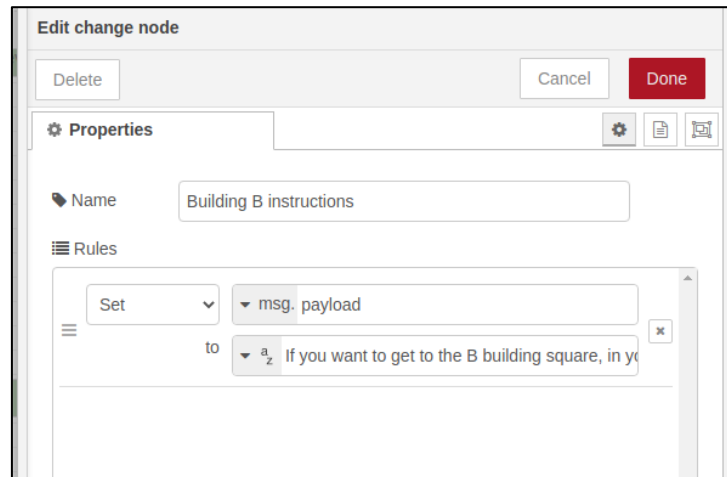


Figure 29 : Change node 1.

4.3.4: ULTRASONIC SENSORS AND BUZZER INTERFACING:

As the name indicates, ultrasonic sensors measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception.

In this system, we use the HC-SR04 which is an ultrasonic ranging module that provides a 2cm to 400cm non-contact measurement function. The ranging accuracy can reach 3mm and the effectual angle is $< 15^\circ$. The distance can be calculated with the following formula:

$$\text{Distance } L = 1/2 \times T \times C$$

where, L is the distance, T is the time between the emission and reception, and C is the sonic speed. (The value is multiplied by 1/2 because T is the time for the go-and-return distance).

In this system, we use ultrasonic sensors to warn the blind person about the surrounding objects by tell him/her warning commands as a sound and alter him/her by turn on the buzzer. However, this process done according to the flow that shown in Figure [30] :

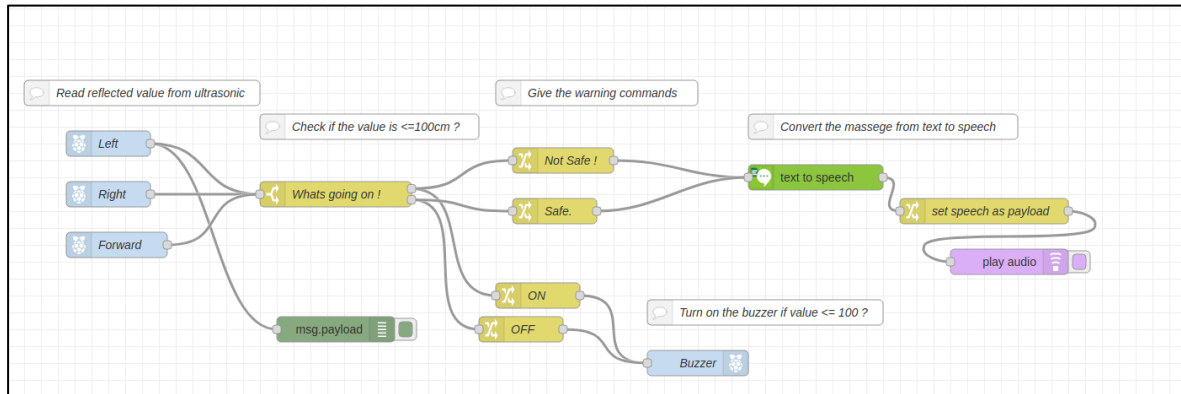


Figure 30: Ultrasonic Sensors and Buzzer.

- ◆ Pi-srf node: Used to read the range from the SRF04 ultrasonic range sensor. It only works with Raspberry Pi. We use three such nodes for each sensor and set up the node properties by selecting the Trig & Echo pins for each sensor and setting the repeat interval. Where the *Left* sensor's Trig & Echo pins are connected to 3,5 Raspberry PI pins, the *Right* sensor's Trig & Echo pins are connected to 8,10 Raspberry PI pins, and the *Front* sensor's Trig & Echo pins are connected to 11,12 Raspberry PI pins respectively. The setting up of the Left one is shown in the Figure [31]. Note that the same setting done to other 2 nodes with change the Pins number and the Name.

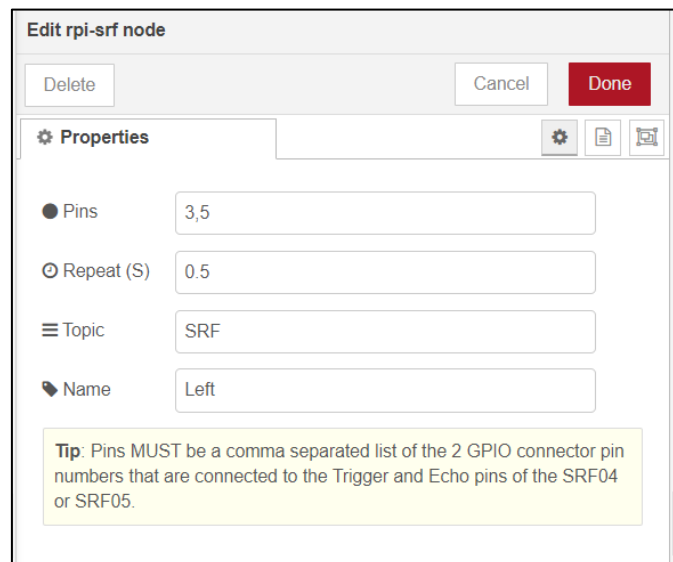
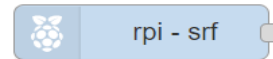


Figure 31 : rpi-srf node.

- ◆ Switch node: in Figure [30], we have used this node to check if the returned value from ultrasonic is ≤ 100 to send warning commands to the blind. Otherwise, to tell the blind that he's safe –he can go ahead-. The properties setting is shown in the Figure [32] :

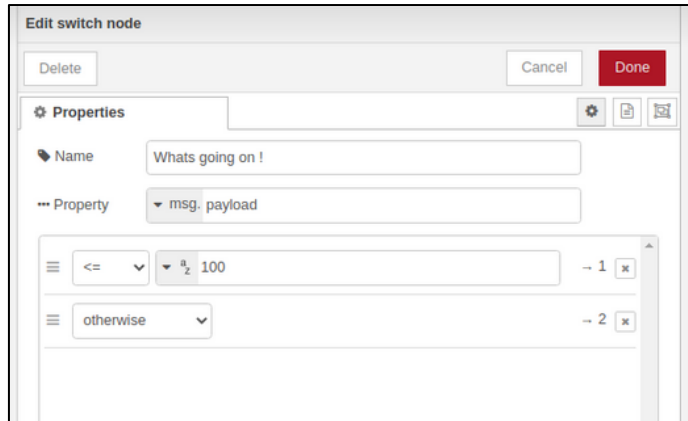
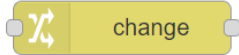


Figure 32 : Switch node 2.

- ◆ Change node: in the flow that shown Figure [30], we have  used this node for three purposes, first, to set the msg.payload to string that contain the warning commands which will be sent later to the text-to-speech node, second, to control the buzzer state, third, to set the msg.payload that coming from text-to-speech node to speech. Therefore, we used five change nodes two for the warning commands (i.e., Safe and Not Safe State), and two nodes for the Buzzer state, and the remaining one node for converting msg.payload to speech.
 - The properties setting for *Safe & Not Safe States* are shown in Figure 33. However, the Safe node its output will hold a message as "You're safe, keep going", where the Not Safe node its output will hold a message as "Be careful, there are some things around you at a distance of less than 100cm".
 - The properties setting for the *Buzzer state* (i.e., ON or OFF) are shown in Figure [34].

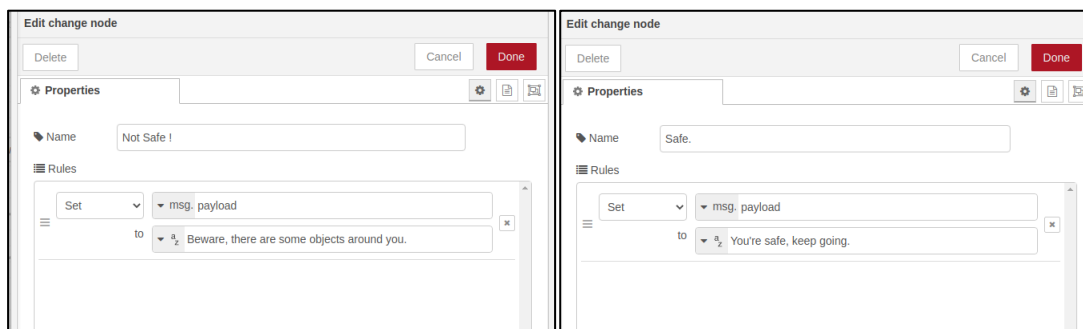


Figure 33 : Safe & Not Safe Change Nodes Setting.

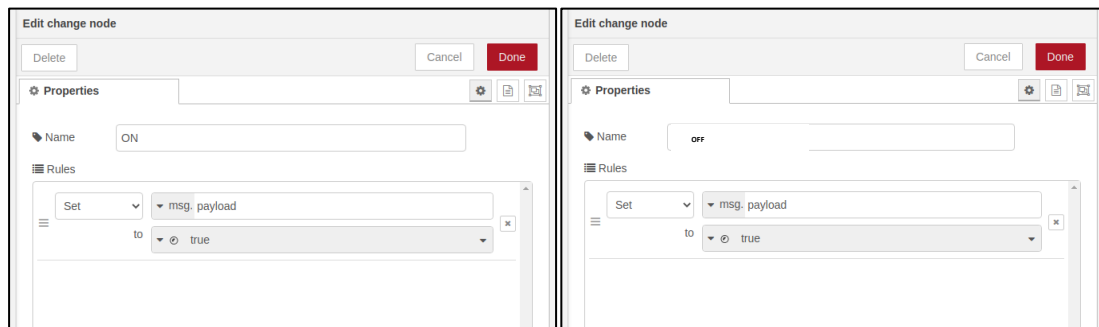
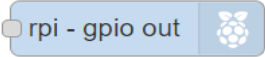


Figure 34 : Buzzer State Change Nodes Setting.

- ◆ Additionally, to convert the warning commands to speech we have used the Watson text to speech node ,and a change node that converting the msg.payload to speech. After that, the sound will be played by play audio node.
-  rpi_gpio out: Raspberry Pi output node. Expects a msg.payload with either a 0 or 1 (or true or false). Will set the selected physical pin high or low, depending on the value passed in. We used this type to connect the buzzer that will be triggered if the return value from ultrasound ≤ 100 to alert the blind. Note that, it will work concurrently with the play audio node (i.e., warning commands). The properties setting is shown in figure [35].

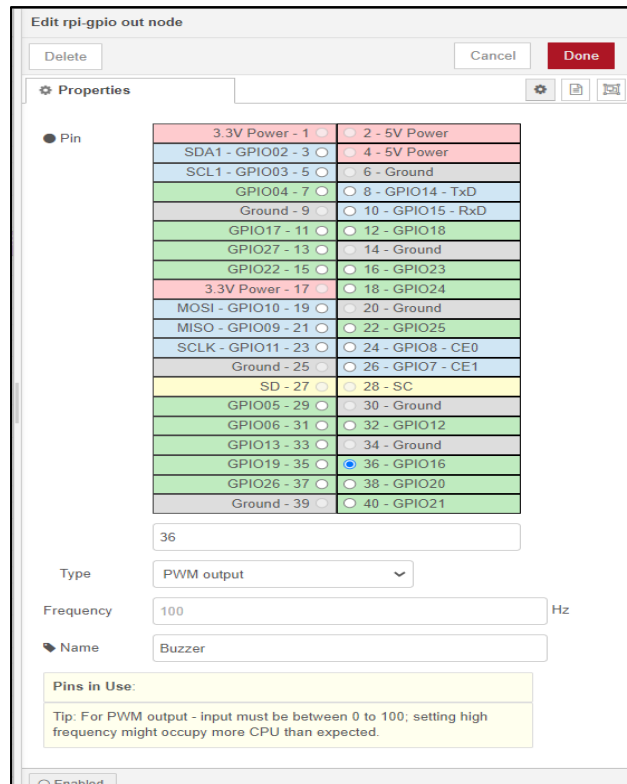


Figure 35 : rpi_gpio out node setting.

4.4: JSON FILE:

With Node-RED as a programming tool, most of the code is automatically generated based on the flow we create. Thus, below we put a snapshot of the file because it too large so the code looks like this:

```
[ {
  "id": "9816f2fd.1357b",
  "type": "tab",
  "label": "Flow 2",
  "disabled": false,
  "info": ""
},
{
```

```
"id": "39bd496c.013e86",
"type": "switch",
"z": "9816f2fd.1357b",
What's going on !", " : " "name
"property": "payload",
"propertyType": "msg",
"rules": [
  {
    "t": "lte",
    "v": "100",
    "vt": "num"
  },
  {
    "t": "else"
  }
],
"checkall": "true",
"repair": false,
"outputs": 2,
"x": 390,
"y": 260,
"wires": [
  [
    "2b8846ac.91305a",
    "50a52cbe.d73f14"
```

```
    ],  
    [  
        "d2da1b82.0b6088",  
        "b70a8be9.2f7918"  
    ]  
]  
},  
{  
    "id": "2b8846ac.91305a",  
    "type": "change",  
    "z": "9816f2fd.1357b",  
    "name": "ON",  
    "rules": [  
        {  
            "t": "set",  
            "p": "payload",  
            "pt": "msg",  
            "to": "true",  
            "tot": "bool"  
        }  
    ],  
    "action": "",  
    "property": "",  
    "from": "",  
    "to": "",
```

```
"reg": false,
"x": 630,
"y": 380,
"wires": [
  [
    "448b0fcc.d3a3a"
  ]
],
},
{
  "id": "d2da1b82.0b6088",
  "type": "change",
  "z": "9816f2fd.1357b",
  "name": "OFF",
  "rules": [
    {
      "t": "set",
      "p": "payload",
      "pt": "msg",
      "to": "false",
      "tot": "bool"
    }
  ],
  "action": "",
  "property": "",
```

```
"from": "",
"to": "",
"reg": false,
"x": 610,
"y": 420,
"wires": [
  [
    "448b0fcc.d3a3a"
  ]
],
{
  "id": "448b0fcc.d3a3a",
  "type": "rpi-gpio out",
  "z": "9816f2fd.1357b",
  "name": "Buzzer",
  "pin": "36",
  "set": "",
  "level": "0",
  "freq": "100",
  "out": "pwm",
  "x": 820,
  "y": 460,
  "wires": []
},
```

4.5: SUMMARY:

In this chapter, we describe system hardware and software implementations, and how we designed our system using the Node-RED platform to build the system flow, leveraging its integration with IBM Watson and Tesseract OCR Engine services. In the next chapter, we will describe how system implementations work to achieve our goals.

CHAPTER 5: VALIDATION AND TESTING:

This chapter shows the testing of the components and the results.

5.1: IMAGE PROCESSING TEST:

Essentially, image processing in this project is implemented by using the Tesseract Open-Source OCR Engine in conjunction with the build-in Node-RED deep learning micro-services from the Model Asset eXchange. Therefore, the object detector node will analyze the contents of the image, and the image caption generator node will generate the caption of the captured image as much as possible and if it cannot generate a caption its output will be null. Thus, we tested it by trying some online images that we can access using the HTTP request node and putting the HTTP address of the images in the node setup properties. So, when deployed the flow, we got the result shown in Figures [36] & [37].

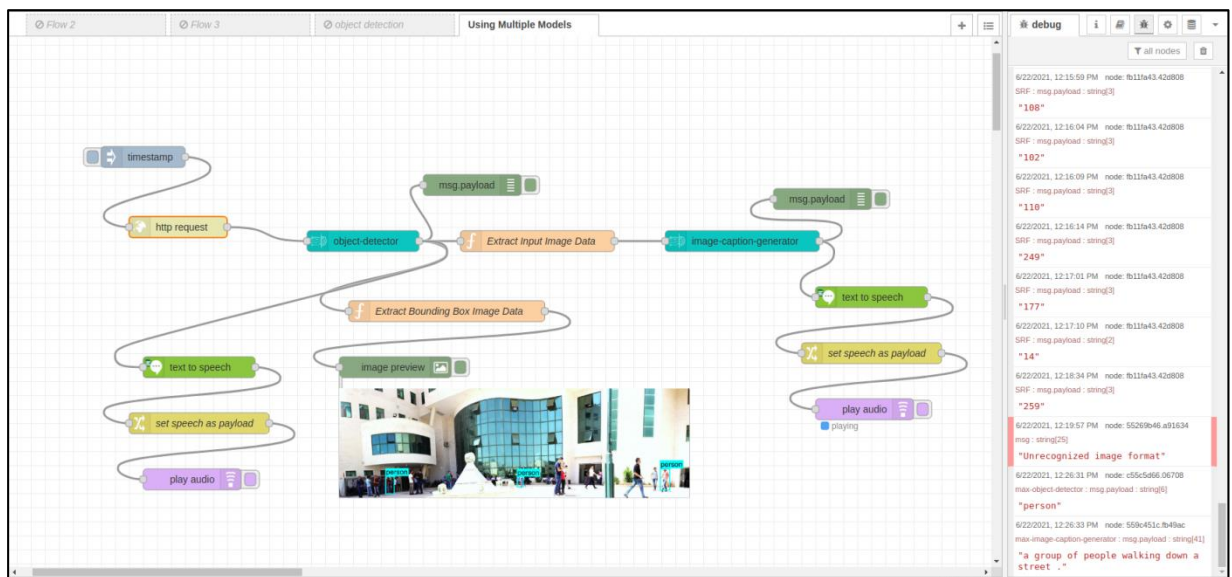


Figure 36: Image processing testing 1.

In Figure [36], we tested an image of the University C building square, and we got from the *object-detector* node the result that shown in *image preview* node that shows drawing bounding boxes around the detected objects -person-. On other hand, the debug window shows the caption as "a group of people walking down a street" that generated by the *image-caption-generator* node where this text caption out as speech by using *text-to-speech*, and *play audio* nodes.

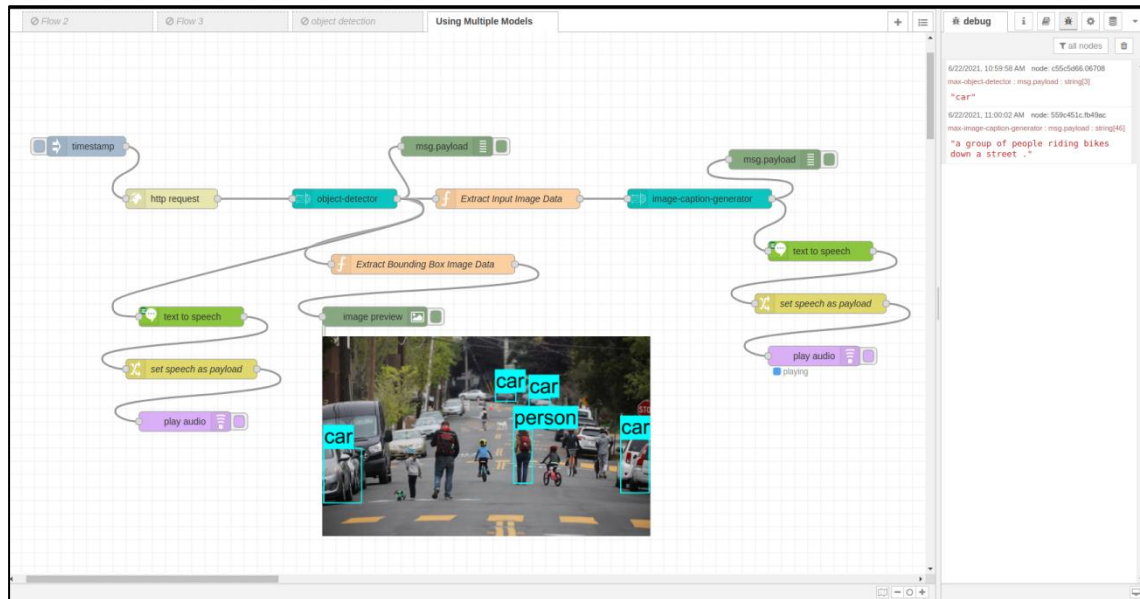


Figure 37 : Image processing testing 2.

In Figure [37], we tested another image and we got that the detected objects are *person* and *car*. While, the caption is "a group of people riding bikes down a street" which is almost true.

In conclusion, the system performs this task successfully and detects the objects.

5.2: SYSTEM'S COMPONENTS TESTING:

5.2.1: Ultrasonic Sensors & Buzzer Testing:

We tested the ultrasonic sensors, which determine the distances of any object near the blind person, so there are three sensors left, right and front that take the result and send it to the Raspberry Pi and then alert him by the buzzer

We tested the three ultrasonic sensors based on their feedback values. Where when we deployed the flow the readings of them appeared in the debug window. However, the left ultrasonic reading will be appeared first, then after 5s the readings of the Right ultrasonic will be appeared, and after 10s the readings of the Front ultrasonic appeared (see Figure below). Also, when the reading values be less or equal to 100cm the buzzer turned on otherwise turned off. And the warning commands played -by the play audio node that plays the speech which came from the text to speech node-

which can be heard from the headphones. The testing result of ultrasonic sensors and buzzer is shown in Figure [38] (See red box).

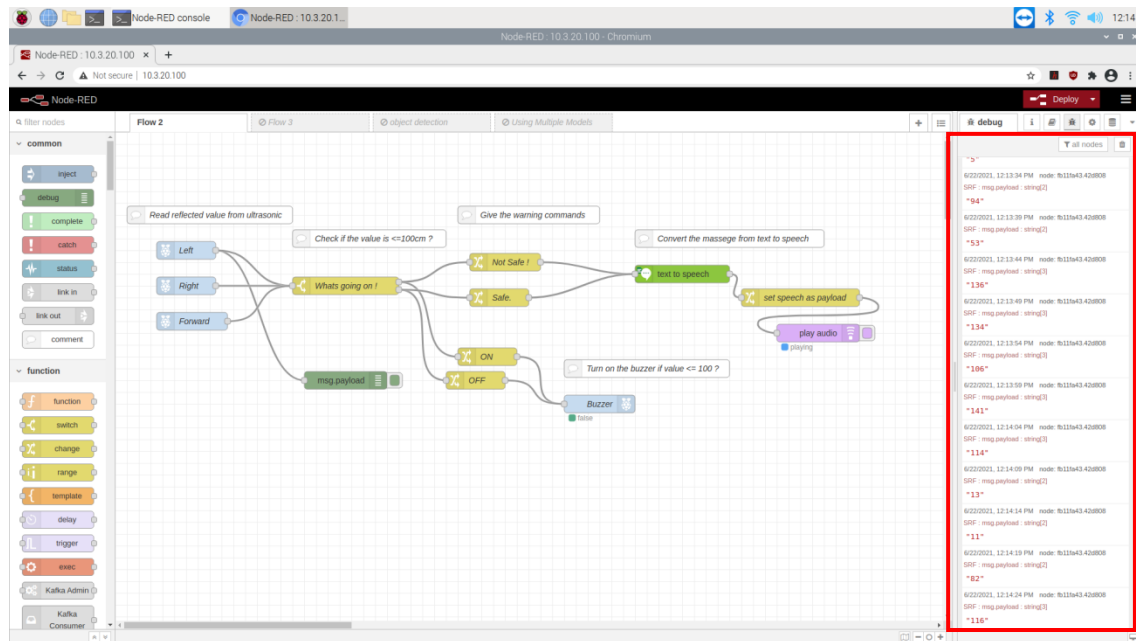


Figure 38 : Ultrasonic Sensors & Buzzer Testing.

5.2.2: CAMERA PI TESTING:

We tested the camera by making the camera take a picture of the street on the right-hand side so that it could capture one of the University streets labels that we were added before, where the picture will be saved in the Raspberry PI Picture Directory. Then, using the object detector node, the object in that image was detected, and also using the caption generator node, the caption was successfully created. Meanwhile, Tesseract OCR was scanning if there were any characters in that image. On the other hand, a warning command and movement instructions were sent via headphones to warn the blind of what was around him and the way to go.

Hence, we got the result shown in Figures 39, 40, and 41 which defines the objects detected and scanned characters by the system in the image:

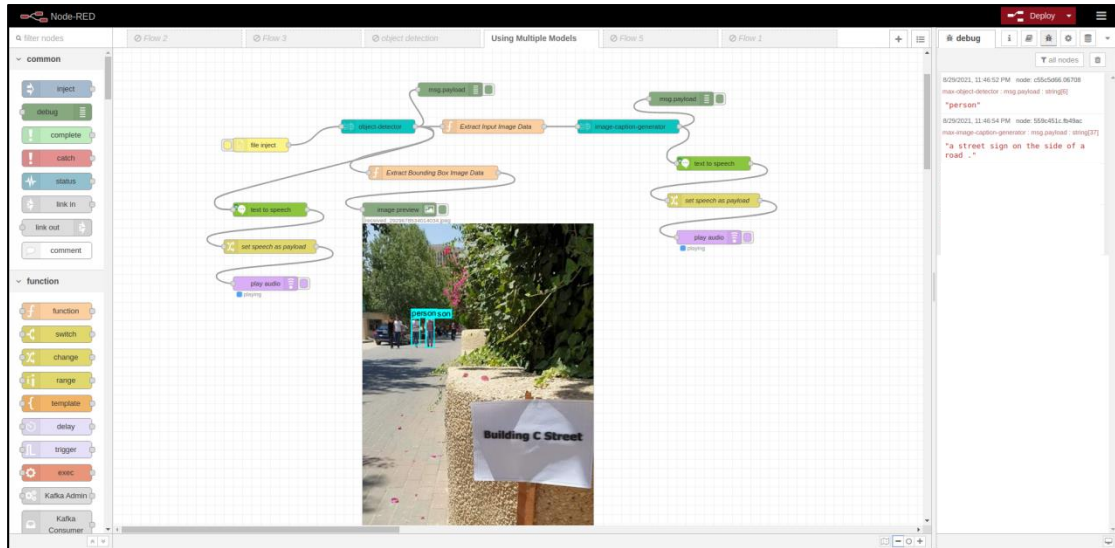


Figure 39: Camera testing 1.

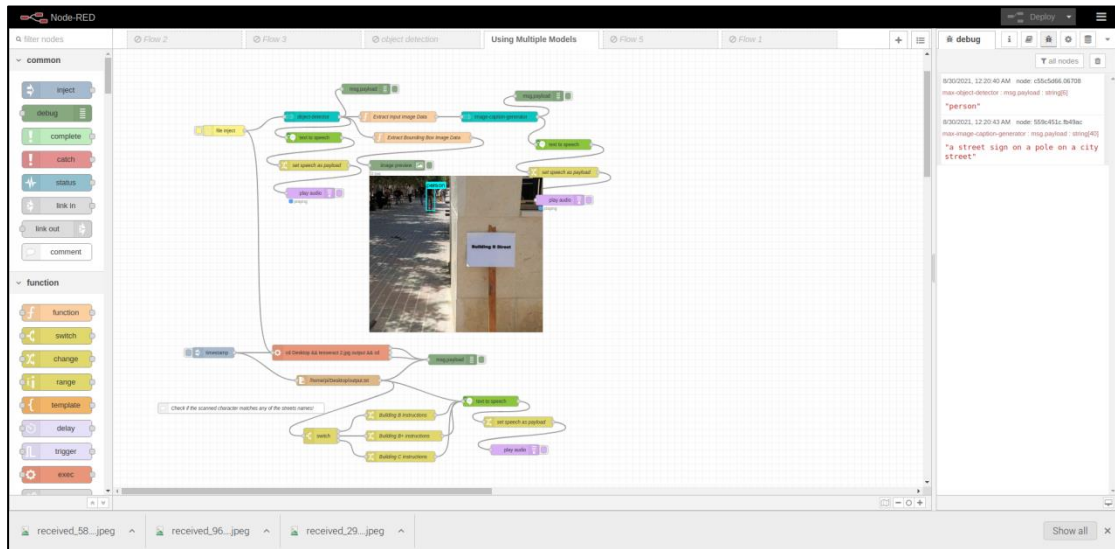


Figure 40: Camera testing 2.

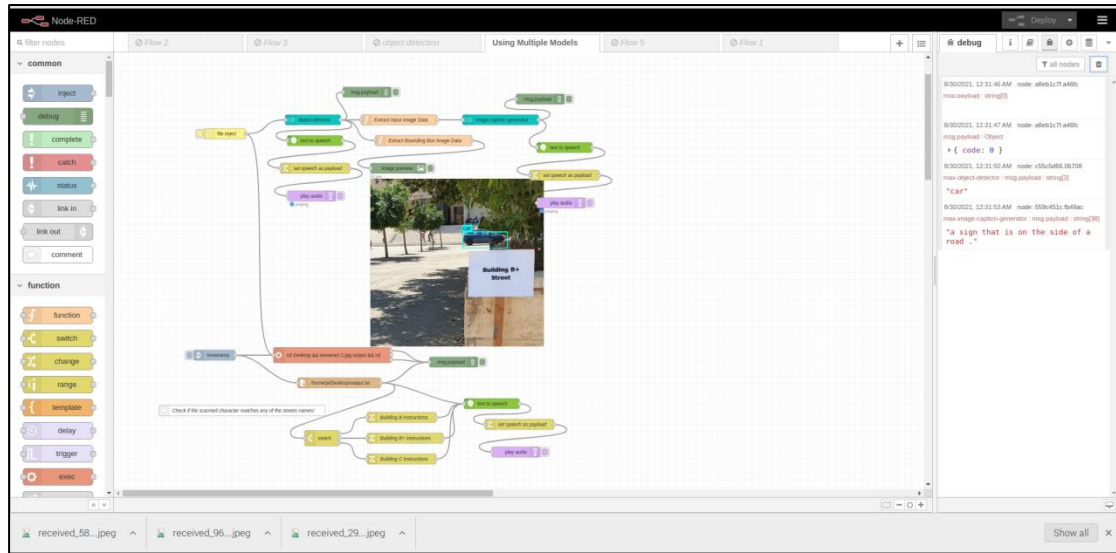


Figure 41: Camera testing 3.

CHAPTER 6: CONCLUSION

In conclusion, in this project, we improved the blind stick in purpose to help the blind to reach the place they want as possible without the help of people. With this stick, the blind can avoid obstacles and danger, receive warning commands and movement instructions based on the images that were taken and the reading values of the ultrasonic sensors. Thus, the stick will provide movement guidelines to the blind, so they will support themselves, and will give them a sense of independence which will help them to live a normal life.

6.1: CHALLENGES:

The main challenges we faced in this project are, firstly, the new programming tool we used to control the stick (i.e., Node-RED) which is not popular among us as colleagues so we couldn't find an expert to help us. Secondly, the IBM Watson Visual Recognition service has been discounted and the creation of any new instances is prevented. In addition, at the beginning of this semester we cannot meet to work on the project due to the lockdown measures during the COVID-19 pandemic. Furthermore, the political situation in the country, and the aggression against Gaza. The third challenge is that someone in our group has experienced some health and personal conditions. Although, we did our best to complete this project.

6.2: FUTURE WORK:

We're looking to make this stick help blind people in all aspects of their lives by training it to analyze a wider range of objects. Also, adding more services to improve it.

REFERENCES:

1. What's IBM Watson? " <https://www.ibm.com/watson>", Dec 2021.
2. D. O Kehinde, O. D .Alao , A.O. Adejugba, Y. A. Mensah , O.M Somefun , I.E Afonne. (2016). PATHFINDER FOR THE BLIND (pp. 1-5). International Journal of Scientific & Engineering Research. Retrieved from "[PATHFINDER FOR THE BLIND \(ijser.org\)](https://www.ijser.org)".
3. Marion A. Hersha, Michael A. Johnson. (2010). A robotic guide for blind people. Part 1. A multi-national survey of the attitudes, requirements and preferences of potential end-users (pp. 1-13). Applied Bionics and Biomechanics. Retrieved from "<https://downloads.hindawi.com/journals/abb/2010/252609.pdf>".
4. Prof. H.S.Hemane, Arundhati Singh, Sneha Shree, Sagar Pandey. (2019). RASPBERRY PI BASED SMART NAVIGATION SYSTEM FOR BLIND PEOPLE (pp. 1-4). International Research Journal of Engineering and Technology. Retrieved from "[IRJET-V7I7575.pdf](https://www.rijet.org)".
5. Juan M. R. A. (2008). Automation and Robotics. I-Tech Education and Publishing.
6. Digital Image Processing, "<https://www.tutorialspoint.com>", Dec 2020.
7. Images, "<https://happycoding.io>", Dec 2020.
8. New in Image Processing, "<https://www.wolfram.com>", Dec 2020.
9. Digital image processing. "<https://en.wikipedia.org>", Dec 2020.
10. Node-Red, "<https://en.wikipedia.org>", Dec 2020.
11. What is the IBM Cloud platform? , "<https://cloud.ibm.com>", Dec 2020.
12. What is a Raspberry Pi? "<https://www.raspberrypi.org>", Dec 2020.
13. Tesseract OCR,"
https://nanonets.com/blog/content/images/2019/11/ocr_flow.png ", Aug 2021.
14. Raspberry Pi,
"https://en.wikipedia.org/wiki/Raspberry_Pi#Performance", Dec 2020.

15. Raspberry Pi

photo "https://cdn.shopify.com/s/files/1/0053/4721/3361/products/RP-RASPBERRY-PI-4-B-1GB_Raspberry_Pi_4_1200x1200.jpg?v=1572920411".

16. Node-Red platform photo,

"<https://developer.ibm.com/developer/default/tutorials/i-running-node-red/images/image001.png>".

17. Ultrasonic Distance Sensor photo, " [https://encrypted-](https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQ5zVD0EhT6O9TZmBa65oLORgP2igfUTtf-xw&usqp=CAU)

[tbn0.gstatic.com/images?q=tbn:ANd9GcQ5zVD0EhT6O9TZmBa65oLORgP2igfUTtf-xw&usqp=CAU](https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQ5zVD0EhT6O9TZmBa65oLORgP2igfUTtf-xw&usqp=CAU) ".

18. Buzzer image, "

https://cdn.shopify.com/s/files/1/2640/3158/products/Squishy-Circuits_Piezzo-Buzzer.jpg?v=152572314 ".

19. Raspberry Pi Camera image, " <https://www.dexterindustries.com/wp-content/uploads/2015/07/Raspberry-Pi-Camera-800x800.jpg> ".

20. A comprehensive guide to OCR with Tesseract, "

<https://nanonets.com/blog/ocr-with-tesseract/#introduction> ", Aug 2021.

21. Object Detection Guide, " <https://www.fritz.ai/object-detection/> ", Aug 2021.

22. Object Detection Example, "

https://www.fritz.ai/images/cats_and_person.jpg", Aug 2021.