ABU DHABI UNIVERSITY

Assistive Technologies for the People of Determination on Smart Home Hubs with AI-based Voice User Interfaces

by

Omar Altawil, Ahmad Hashi, Mahmoud Bakir, Omar Aoun

A thesis submitted in partial fulfillment of The
Requirements For The Degree of Bachelor in Electrical and Computer Engineering

in the

COLLEGE OF ENGINEERING
DEPARTMENT OF COMPUTER ENGINEERING

December 2018

In The Name of Allah, The Most Benecent, The Most Merciful.

Declaration of Authorship

I, AUTHOR NAME, declare that this thesis titled, 'THESIS TITLE' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Signed: _____

Signed: _____

Signed: _____

Date: _____

"He who walks in the path of seeking knowledge, GOD will guide him to the path of paradise."

Holy Prophet Muhammad (Peace Be Upon Him)

ABU DHABI UNIVERSITY

Abstract

COLLEGE OF ENGINEERING
DEPARTMENT OF COMPUTER ENGINEERING

Bachelor of Computer Engineering

by Omar Altawil, Ahmad Hashi, Mahmoud Bakir, Omar Aoun

In this project, we implemented a smart system that has a sole purpose of helping the Visually Impaired People Of Determination in leading an easier and better lifestyle. Our proposed system had an AI and Voice User Interface base. The project consisted of three main sub-systems, the first part was a navigation skill that the Visually Impaired People Of Determination can use to navigate and move independently and safely within that certain environment. The second part was was a skill that gathers information around the clock in the surrounding environment and alerts the user whenever he or she asks about these indicators (such as temperature, humidity, number of people in the room). The third part was a skill that can be used for online banking purposes, from banking details to online shopping with extreme ease and guaranteed security.

Acknowledgements

We give all of thanking and gratitude to at the beginning to Allah Almighty, as everything is subjected to the will of The All Might Allah, Praise be to him.
We would like to thank all of our dear family members and friends for their extraordinary emotional support thought out the whole project. Next, we would like to give especial thanks to the department of computer engineering, as they provided great help since the beginning and thought out all these four years. Without them, this project would havent being possible.
Next, we would like to thank Engineer Yasmina Al Khalil for her assistance to us thought out the first phase of the project, Engineer Yasmina made sure to give us all the necessary guidance and advices. She also helped greatly in providing 90 percent of the parts that were needed in order to build up our system, as she gave us some parts and directed us to where we can find the rest of the components of our system. She also helped us code wise as she gave comments about different libraries and SDKs.

Next, we would like to thank Engineer Marah Al Halabi as she helped us in the phase of our project. Engineer Marah helped us with our 3D modeling and our PCB design as she give us her feedback and how can we scale the 3D design, she also helped in the PCB design and gave some advices on that regard.

We also would like to thank New York University and their Innovation lab assistance for their help and positive feedback.

Finally, we would like to give thanks to Doctor Mohammed Ghazal our supervisor and Associate Professor of computer engineering; a person whos responsible of the successes of our project. Doctor Ghazal was along us thought out the whole journey and from the beginning. He provided us with constructive advices about our design and our implementation of the whole project. Also, Doctor Ghazal helped us a lot with debugging and overcoming all the different difficulties that we have faced thought out the whole project. Not to mention that face that he was extremely open minded in all of our meetings where he occupied his busy schedule to conduct our meetings that were for the more part long and time consuming. Its save to say that this project could have not been completed without Doctor Ghazal, Big thanks to him.
...

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Statement

Ever since the Industrial Revolution in the late 18th century, the development of technology has been taking huge leaps and in an extraordinary bases as great inventions such as motors and great discoveries such as electricity changed forever the course of human life on planet earth. Even now in the 21th century, the rate of development of technology seems to raise in an exponential way, but there are some things that are yet to be improved using modern technology. Nowadays, with the use of advance medicine, illnesses that were once deadly are now considered light weighted, yet some illnesses are still considered permanent even with all the improvement that we have in modern medicine. Our focus is on the people of determination especially the visually impaired people of determination. According to the World Health Organization, there are approximately 1.3 billion people live with some form of vision impairment [4]. The worlds blind will increase threefold from about 36 million today to 115 million in 2050 as populations expand and individuals grow ever older, researchers said Thursday. The number of people with a moderate to severe vision impairment only those not corrected by glasses, contact lenses or an operation will also near triple, from about 217million to 588 million over the same period [5]. Although technology has been used to help easing the life of people in all of its various aspects, the community of the visually impaired people of determination got the least share of technology when compared to others. His Highness Sheikh Muhammad Bin Rashid Al Maktoum the Prime Minister of the UAE and the governor of Dubai Emirate have said: "The true disability is of a person is when that person is not able of moving forward and rather stay in the same spot. The prove to the extraordinary power of the human will is shown in the people of determination as they faced forcefully all the challenges that life putted them through and yet they have

reached their goals and succeeded in their lives" [6]. As we can see from the bar chart

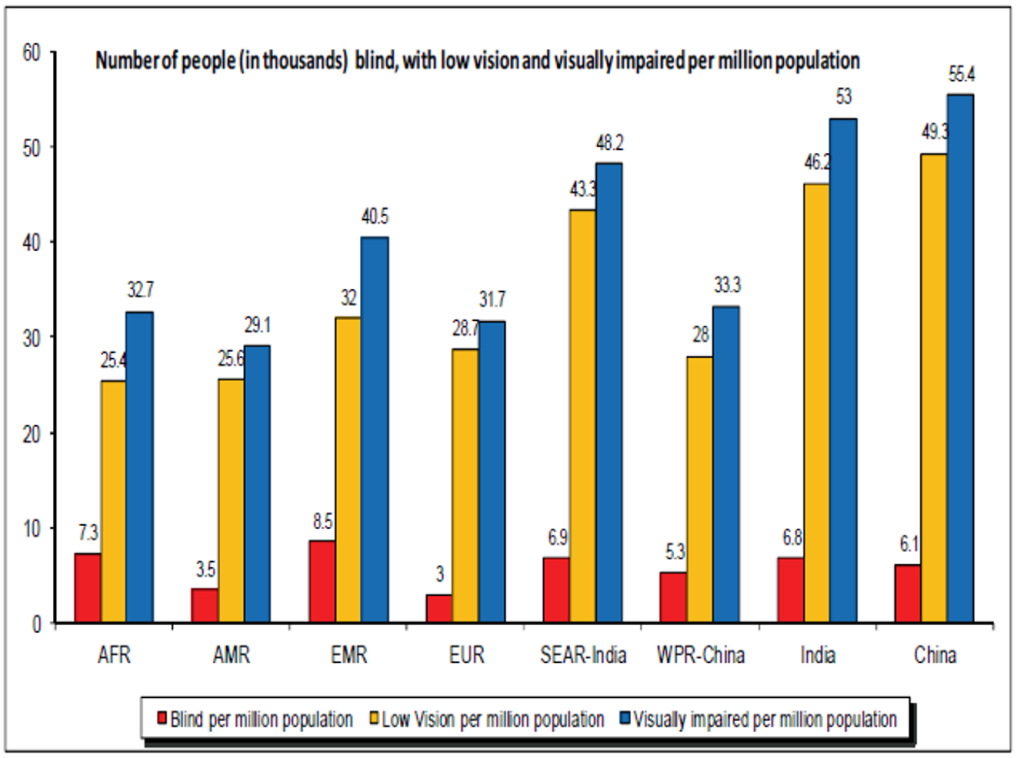


FIGURE 1.1: World Health Care Statistic

here, which shows some statistics about the visually impaired in 2010 the problem of low and no vision worldwide is serious, and yet we dont see that much interest in solving this particular problem or even helping in reducing its effect around the world in the technical society. Visually Impaired People Of Determination dont have their custom made technical devices that aims to help them lead a more easier life. Although they should have the priority in such fields, people tend to seek their own comfort and forget about the Visually Impaired People Of Determination as technology advance and we see new innovations on how can we make our live easier ignoring the people how need such technologies the most.

1.2 Motivation

Our main motivation in this project is to shed some light on the importance of the Visually Impaired People Of Determination in our society and how all the technical society is responsible for not providing new and powerful devices that helps the Visually Impaired People Of Determination in their every day to day live activities. From that point, we started our projects idea which was crafting our own smart home system that is custom made for the Visually Impaired People Of Determination. For that, we have used

the power of two concepts; IOT and AI. We made our system using smart sensors, smart speakers and a Camera along with the use of deep learning algorithms. We implemented everything considering the fact that our target user is a person that is visually impaired. Our system can be divided into three main parts, firstly we have designed a navigation system that uses a camera to navigate the visually impaired throughout their lives, plus it gives a detailed description of the surrounding environment of the user. Secondly, we designed a system that will tell the user about important events that are happening in a room. Finally, we thought about how its difficult for the Visually Impaired People Of Determination to go to a bank or use online/mobile banking, so we have designed a system that will allow the user to access their banking information using natural voice processing.

1.3 Background

1.3.1 Google Cloud(Vision API)

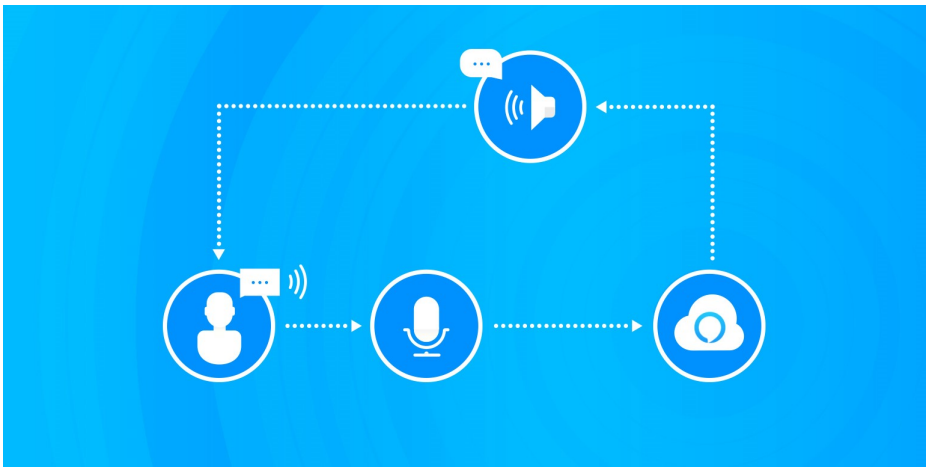
Google Cloud Platform(GCP) is a platform where users can use Google's cloud and infrastructure as their own. Google clouds vision API is a strong API that allows the camera to have visual recognition of anything like, labels, faces, objects, landmark and many more.

Figure 1.2: GCP

1.3.2 Alexa Voice Service

Alexa Voice Service (AVS) is Amazon's suite of services built around its voice-controlled AI assistant for the home and other environments. AVS and Alexa were first introduced with Echo, the company's intelligent speaker, which enables voice interaction with various systems in the environment and online. Alexa is available for an ever-increasing number of other devices, including smart phones, tablets and remote controls.[7]

Alexa Voice Service (AVS) enables you to access cloud-based Alexa capabilities with the support of AVS APIs, hardware kits, software tools, and documentation.[8]



AVS is Integrated with Amazons system, this helps to access amazon directly from the AVS. Other than that AVS is used in smart homes to connect smart appliances together, another advantage of AVS is to control the smart ecosystem in a home like controlling smart lights, thermostat, smart TVs and much more. The best thing about it is that it is a free service for developers to test, develop it's skills.
The AVS can get a lot of things done like:

- play music and audio books
- Play movies on the TV
- Order food
- Search for information

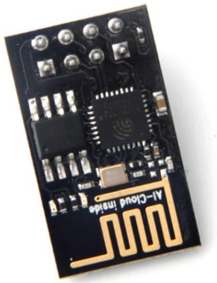
AVS is a really great system to work with, that is because the applications of it is limitless, it can be used to do basically anything that the developer wants it to do.

1.3.3 Jovo

Jovo development framework for cross-platform voice apps that let you with one codebase build voice apps for Amazon and Google Assistant with AI that help building professional apps.Before a user's speech input is reaching the Skill.

1.3.4 ESP01

The ESP8266 ESP-01 is a Wi-Fi module that allows micro-controllers access to a Wi-Fi network.[9] The ESP01 is a SOC(System on a Chip), so it can still be programmed to act as a micro-controller, which makes it versatile.



Arduino IDE is used to program the ESP01, since it originally used as a Wi-Fi module, it can do thing that other arduinos can like connect to a database or control devices using the Wi-Fi, which makes it more suitable to be used the projects that requires Wi-Fi.

The disadvantage of the ESP01 is that, it has only eight GPIO since it is a module, which makes it a bit inefficient to be used as a micro-controller, this means that it is more suitable to be used in a small scale projects but bad at medium to large scale projects.

1.3.5 Github

GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 31 million developers.[10]
Github is an amazing platform that allowed us to save, share, and get code with the community that they have built over the years.

FIGURE 1.3: GitHub

1.3.6 VS-Code

Visual Studio Code(VS-Code) is a code editor made by Microsoft that supports all operating systems, it has an editor for all programming languages, and it is the go to editor for many programmer through out the world. It also supports debugging, embedded Git control and many more.

FIGURE 1.4: VS-code

1.3.7 Fusion 360

Fusion 360 is a free application that includes CAD, CAM and CAE tools. Fusion 360 is used to create and design 3D models in anyway the user wants. Fusion lets the user to sketch, design, modify and assemble different components to create one wholesome model.

Fusion 360 is a product that is made by autodesk. Autodesk is a software company that sells many products that can be used for manufacturing, architecture, entertainment, media, construction, and building.

FIGURE 1.5: Fusion 360

1.3.8 Fritzing

Fritzing is an open-source hardware initiative that makes electronics accessible as a creative material for anyone.[11] Fritzing allow users to design and build hardware circuit and PCBs.

FIGURE 1.6: fritzing

1.3.9 MRAA and UPM

For any Iot device to connect to is sensor you need a code ,so you need MRAA and UPM.MRAA is Low Level Skeleton Library for Communication on GNU/Linux platforms.MRRA does not limit you to specific hardware because it is compatible with all hardware with a small modification[12].UPM is a framework for developers with high-level APIs that make the connection to sensors and actuators easier in Iot solutions

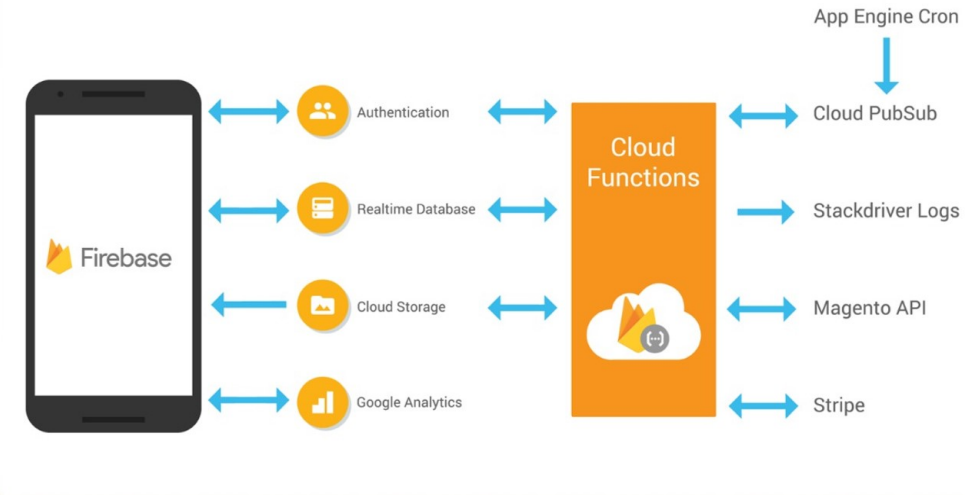
The UPM repository provides software drivers for a wide variety of commonly used sensors and actuators.Developers can access the interfaces for each sensor by including header file and the class for the associated sensor.UPM and MRAA support Multi−Language like: C/C++, Java*, Node.js∗ and Python. Most sensor and actuator is supported [13].,for instance:

- Temperature sensor
- Light controller
- Gas sensor
- Light sensor
- Humidity sensor
- Pressure sensor
- Analog to digital converter

1.3.10 Amazon Developer Portal

Amazon Developer Portal includes tools for developers to build apps for Android and iOS mobile devices.It also, has console that has tools and services that enable developer to add skills for your project ,test the skill,and publish it.In order for the user to access it he need to create Amazon account.

1.3.11 FireBase



Firebase is an application development platform that supports web development and mobile development for both Android and IOS. Firebase is a scalable Real-time backend (BAAs) which functions as a Real-time Database [14]. Firebase started as a startup company named Envolve by James Tamplin and Andrew Lee in 2011 and it gave developers an API that can integrate online chat functions into their websites. After some time, the owners decided to individualize the real-time architecture that powered their chat

service which led to the birth of Firebase; a separate company in 2012.By 2014, Firebase was acquired by Google, and in 2015 Google acquired Divshot which got merged with Firebase, only after that Firebase services became a unified platform for mobile developers [15]. Unlike other databases which uses HTTP calls to synchronize data, Firebase uses a WebSocket which is very fast comparing to HTTP calls were the WebSocket speed depends on the clients network speed which enable Real-time update of the database whenever changes occur. Beside the fact that Firebase provides a Real-time Database, Firebase also provides other useful services to its which are the following:-

- Cloud Firestore
  Cloud Firestore is a NoSQL document database that simplifies the processes of storing, sync, and query data for both your mobile and web applications at global scale. It integrates with other Firebase services to make the users application nearly server-less [16].

- Cloud Functions
  Cloud Functions is a service that takes care of the configuration, scaling and even adding a new server or removing an existing one depending on the usage pattern of your application by writing a single JavaScript function based on a desired event [17].

- Authentication
  Firebase Authentication makes it very easy to log in to your application by providing a lot of methods of Authentication were it has a built-in functionality for a third-party providers like Facebook, Twitter, Github and Google which makes the log in process super easy for the user. It also gives the developer the choice of building his/her own UI or use the open source highly customizable Google UI. After the Authentication process is over, the user information is returned back using callbacks in order to give the developer the freedom to personalize his/her application to their specific users were each user info comes alone with a unique user ID and this ID is used to identify the user and his/her access privileges, also it manages the users session status were the user will remain logged in even after closing or restarting the application [18].

- Hosting
  Firebase Hosting is made especially for front-end Web applications and its a static web hosting provider. Firebase Hosting is fast irrelevant of the users place were files which are deployed to Firebase Hosting is cached on SSDs at CDN edge servers scattered all over the globe also, Firebase Hosting automatically configures an SSL certificate for every site it deploys [19].

- Cloud Storage

  Firebase Cloud Storage lets the user upload his/her data to the cloud using Firebase Storage API to make it possible to share with other users. The data transfer happens over a secured connection add to that having a robust data transfer so that it will resume automatically if the device disconnects. The provided API Storage is backed by Google so the fear of running out of storage is not there [20].

- ML Kit

  Firebase ML Kit is a service that brings the power features of Artificial Intelligence and Machine Learning into the hands of the ordinary mobile app developer. ML SDK provides a set of ready to use APIs that covers the most common used mobile application cases such as text recognition, face recognition and image labeling. ML Kit API runs on the device or on the cloud or both. On-device ML Kit API process information at a high speed rate and dose that without connecting to the network were Cloud based APIs use Google Cloud Platform along with ML Kit technology to produce a very accurate result [21].

1.3.12 Angular

In order to understand Angular, we need to talk a little bit about Angular.JS first. Angular.JS is a JavaScript based open source front end web application framework developed by Google and the community of developers [22].On the other hand, Angular is a complete rewrite from Angular.JS and unlike Angular.JS, its a type-script based open source front end web application platform. Angular 2 was the first rewrite of Angular.JS whereas Angular 4 is the newest version of Angular 2 with some improvements and is compatible to it. Angular was also developed by Angular team at Google and by the community of developers and is considered a newer version of Angular.JS. Angular is used to build applications that work on mobile, web and desktop platforms [23]. Before diving into Angular and its components, lets explain the difference between Angular and Angular.JS in terms of Architecture, Language, Expression Syntax, Mobile Support and Routing.

Architecture :

- Angular.JS: Its architecture is based on Model View Controller (MVC) design which is used to describe the behavior of the application and manages its rules, logic and data. The design as the name suggests is made of a Model, View and Controller. The Model has a primary responsibility which is storing data is a way similar to primitive types or objects in Java. Its considered the simplest level where it does not provide Setter and Getter methods [24]. The View as a UI that shows the desired data to the user in a certain format. Last but not least, the Controller which decides which data is to be shown by the View as it responds to the input provided by the end user and triggers the requested operation and shows it. Basically, the Controller orchestrate the interaction between the Model and the View [25].

- Angular: Unlike Angular.Js, Angular doesnt use MVC design instead it uses Components where a Component is a directive with a template [24].

Language:

- Angular.JS: Its written using JavaScript which is described as an interpreted high-level programming language that is characterized by being dynamic, weakly typed, prototype-based and multi-paradigm [26].

- Angular: It uses Microsofts TypeScript which is an open-source programming language developed by Microsoft and its used for application-scale JavaScript [27].

Mobile Support:

- Angular.JS: Does not support mobile development [24].

- Angular: Supports mobile development [24].

Routing:

- Angular.JS: uses the following code to configure routing.

```
routeprovider.when()
```

- Angular: uses @RouteConfig() to configure routing.

Although Angular.JS was mainly developed for designers not developers, when it comes to performance we have to talk about the advantages and disadvantages of both Angular.JS and Angular in order to obtain a fair comparison of the two. Lets start first with Angular.JS:-
Advantages :

- Unit testing ready [24].
- It provides a great MVC data binding that helps in developing Applications much faster [24].
- It uses HTML, which is an intuitive language as a declarative [24].
- It doesnt use any require frameworks or plugins which makes it a very fast Front-End development solution [24].
- Extremely compatible (PC, IOS, Android) [24].

Disadvantages:

- Complications due to having multiple ways to do the same things [24].
- Implementations scale poorly [24].
- JavaScript dependent [24].
- It has a lagging UI when having more than 200 watchers [24].

Now lets look at Angular:-
Advantages

- It has the option of fast development process [10].
- It uses TypeScript which is useful for building huge applications [24].
- Very easy to write Tests [24].
- Offering modularized animation package [24].
- The View Engine now generates less code when its on AOT mode [24].
- Great for using incases of single-page web applications with an extended interface.

Disadvantages:

- It gets slow when it comes to showing large amount of data [24].

In our project, we decided to use Angular since its the newest and was the best fit for our implementation. Angular offers a range of components that can be used to implement common interaction patterns, these components can be categorized under six main titles which are the following:-

1. Form Control
2. Navigation
3. Layout
4. Buttons Indicators
5. Popups Modals
6. Data Tables

Angular Lifecycle Hooks
Almost every Angular component has at least one Lifecycle Hook and the directives under that Angular component follows the components Lifecycle Hook. Angular creates Lifecycle Hooks, renders and further creates and renders its children; checks it when its data-bound properties change and destroys it before removing it from the DOM [28]. Angular provides various Lifecycle Hooks, the figure below shows Lifecycle Hook Methods.

1.4 Literature Review

In the beginning, we did some research to see any other related projects or work to assist the visually impaired people of determination. The first good idea that we have stumbled upon was talking smart glass for the Blind by Amal Shajan. the project is all about helping the visually impaired people of determination, and it uses a few ultrasonic distance sensors, an Arduino Pro Mini, an MP3 player module, and some vibration motors. It basically alerts the person whenever there is an obstacle in front of him /her.Although it's concept is simple, but yet it's a very good idea. Figure ?? shows the glasses [29]. The the second idea we found was a smart blind stick navigator. This

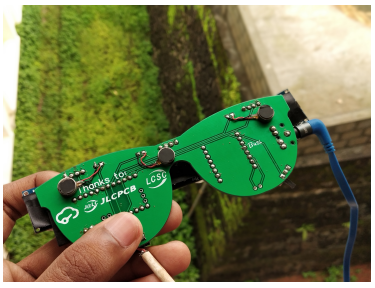


FIGURE 1.7: The Glasses

device provides obstacles notification and GPS location to the guardian and authority via SMS messages. The smart stick concentrates on sensor to get the most accurate location information. Figure ?? shows the stick [30]. The third project that we found

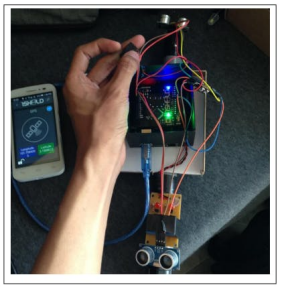


FIGURE 1.8: The stick

was an mobile application that is called Be My Eyes, which is basically an application that connects the visually impaired people of determination with sighted volunteers and company representatives for visual assistance through a live video call. Its a free application and ts available on both Android and IOS App store. Figure ?? shows the application [31].

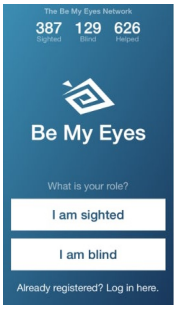


FIGURE 1.9: Be My Eyes

1.5 Impact Statements

1.5.1 On The Society

In our project we have tackled many things to help the society. First thing our smart banking system, this system will help all the people of the world by making the interaction smooth, fast and secure and using only their voice, this saves a lot of time, this will allow the people to send money, order online, check their balances and previous transactions with limited interaction which would be our smart button for the security sense of our system.

Furthermore, our smart vision system will help people who are visually impaired to navigate through their lives by giving an overall description to the rooms that they enter,

plus we have integrated real time description of the area that he has entered using his camera which will also help with their lives, plus the smart vision acts as a security layer in the smart banking system for the visually impaired to tell them if they are alone in the room or not, so that sensitive information would not fall in unwanted hands.
Finally, in our super sensor model this will help people to get real time activity about the room, from temperature, to humidity to many more, plus we have implemented machine learning in the mics to calculate for the user an approximate number of how much water are they using, this will allow the user to be more aware about the water consumption and will try to limit their usage to a minimum so save water.

1.5.2 On The Economy

The project impact on the economy is really good, since in our designs we have focused on simplicity and not using many parts, some of our systems can be used and implemented for a low amount of money. The smart banking system should be used with a voice assistant which can be reached by just downloading the app on the phone, but for maximum results the smart assistant hub should be bought which could be ranged from as low as 200 Dhs to a maximum of 1000 Dhs, in our implementation we have used PCBs, 3D printed design and the smart assistant. As for our vision it will almost cost the same as the smart banking system since it uses almost the same components as it. Furthermore, the supersensor would be a bit more expensive since we are using a unique processor with multiple sensors which cost a bit more than the other two systems but isn't that overpriced for what the user is getting.

1.6 New Technologies Used

Here's a list of all the new technologies that we have used in our project:-

- Respeaker Board
- LSM9DS0
- ESP
- Alexa Voice Service
- Angular
- Firebase

- Arduino IDE
- Putty

1.7 Report Objective

This thesis paper is about our capstone project and it explains everything that we went through in order to produce a functioning system. The project is about helping visually impaired People of Determination using AI-based Voice User Interfaces and Smart Home technologies. The thesis for this project is divided into seven chapters. The first chapter is the introductory chapter where we give a brief idea about the project , who is it for, why is it helpful and some technical details about the project. In the second chapter, we discussed the designing of our smart system and its sub systems and we explained the architecture of the whole project. The third chapter was solely dedicated to talk lengthily about Alexa Voice Service as it was a crucial part of this project. The fourth chapter dealt with the implementation of our smart system where we have shown step by step how did we build our system from point zero to a completely functioning system. The fifth chapter explained the Project Management aspect of our project, the load distribution and the overall team dynamics. The sixth chapter contained our results and discussion where we did our testing and analyzed the produced outcomes of each test. The seventh and final chapter is conclusion and future work and it had our summary of the overall findings in our thesis alone with our future work for this project.

Chapter 2

Design

2.1 Vision

2.1.1 Problem Description

The community of people of determination have a lot of support here in the UAE as Sheikh Hamdan bin Mohammed bin Rashid Al Maktoum Crown Prince of Dubai said: Our society is a place where each and every member of people of determination is supported in every means possible [32]. But, the visually impaired people of determination in particular have yet to fully merge into the structure of the society due to various challenges, for example the sidewalks on the streets are not pedestrian friendly for the visually impaired people of determination. We are yet to overcome the challenge of making navigation throughout our cities, streets and even our own houses much better for the visually impaired people of determination. Considering these challenges, we came up with the idea of vision, an IOT based application that works as a guide to the visually impaired people of determination.

2.1.2 Hypothesis

The system will detect all the event that happened in the room using AI and it is connected to Alexa.

2.1.3 System Architecture

Our idea of the Vision System is to make a system that the visually impaired people can use to overcome navigation problems that they might face in the case of them entering

a new environment. Our idea was to focus on one aspect or in other words one of the many challenges faced by the visually challenged people which is coming to be familiar with a new environment or one that contains quite varying objects in terms of location. So, we came up with our initial would provide the visually challenged person with a brief description of the surrounding environment in no time. The visually challenged person then would navigate inside that new environment depending on the description. We further narrowed down our environment to a place that our target user frequently visit or go to, such as his or her own house or a room within the house. We had a couple of thoughts for implementing our hypothesis, in the beginning we had the idea of using a Natural Speech Recognition system through a Voice Interface System (Alexa) and we thought of manufacturing some sort of a chip that goes inside the front pocket of the T-shirt. The chip consists of Raspberry Pi Zero and a micro camera embedded with the Pi in order to have a functioning unit. This unit functions when the target user enters him or her house, wakes up VIS via voice command Alexa and then asks Alexa to describe the room for him/her, The micro camera will take a photo of the room and its surroundings and it will upload it to Firebase and then link it to Firebases ML Kit where the deep processing of the photo will take a place After the processing of the photo is completed, the results is going to be used to construct a full description of the room and its objects. A second implementation that we thought of in using a mobile phone instead of a chip. This unit works in the sense that the target user will enter in the desired environment lets say his house and as soon as the target enters, he or she can open Vision Application in the phone using voice command and take a photo. The photo will be sent to the Realtime Database of Firebase and then the picture will go under deep processing using ML Kit from Firebase which will result in labels which describes objects in the photo. The description is then linked to the voice interface system (Alexa) through Alexa server and Jovo. The end result is that the voice interface system (Alexa) will describe the room to the target user. After some thinking, we came up with yet another different implementation which was having a Camera positioned in an angle that lets it capture the whole room take care of providing the photo where the system will work when the target user says to the voice interface system (Alexa) Describe the room. The voice interface system (Alexa) will respond by taking a photo, upload it to Firebase and then link it to Google cloud Vision API which will do deep processing, after that we get the result and use it to produce the overall description, where the original description of the room which is saved in the Angular Web Application will be combined with any new or recently made changes in the environment that will be known if it happens from Vision API, which will result in a very accurate description of the environment. Figure 2.1 explains how Vision system will process.

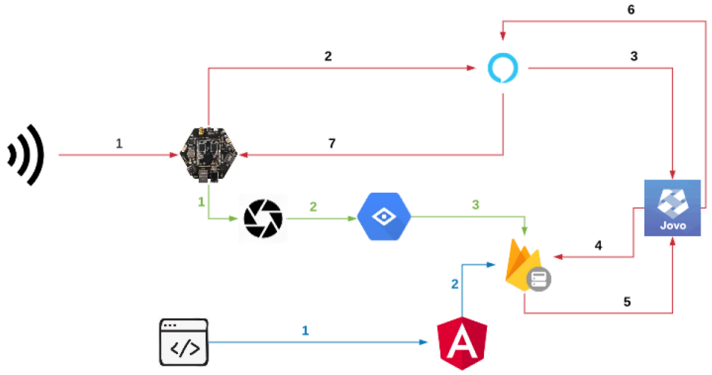


FIGURE 2.1: Vision System Architecture

2.1.4 System Components

The components that are used in order to implement the Vision system are the following:

- Voice Interface system (Alexa)
- Respeaker Board
- Alexa Server
- Jovo
- Angular
- Firebase Realtime Database
- Google Cloud Vision API
- GoPro Camera

2.2 Smart Banking

2.2.1 Problem description

As technology advanced banking systems went from tradition to online to mobile, and now we wanted to implement an AI banking system that is both secure and hands free using IoT and deep learning. This is a tough challenge since many things have to be addressed when implementing a banking system, an example would be smooth experience, fast interaction and the biggest challenge of all security.

2.2.2 Hypothesis

A system that uses natural language to implement a smart bank inside the house with minimal interaction using smart assistants and a smart button.

2.2.3 System Architecture

The smart banking system as mentioned before, is an IoT implemented system that is used to help the visual impaired to access their banking services in an easier and faster and more secure way, this is done by using the voice interface system (Alexa) by giving it commands to control the banking system. The system can do anything from checking the balance, to transferring money, to pay the bills to many more.
To do that first the user will have to login by using their voice, and then after that they can access their banking services, now to make the system even more secure we added a hardware button that is kept by the user to act as a two-factor authentication, this button will be used when the user wants to do an action and this button can be modified to be able to do a combination of presses that suits the user.
We also added a layer of security which is visual recognition using a camera, that is designed to help the visually impaired to not get any sensitive information if there are others in the room.
The architecture of our design and the to intent architecture is shown below:

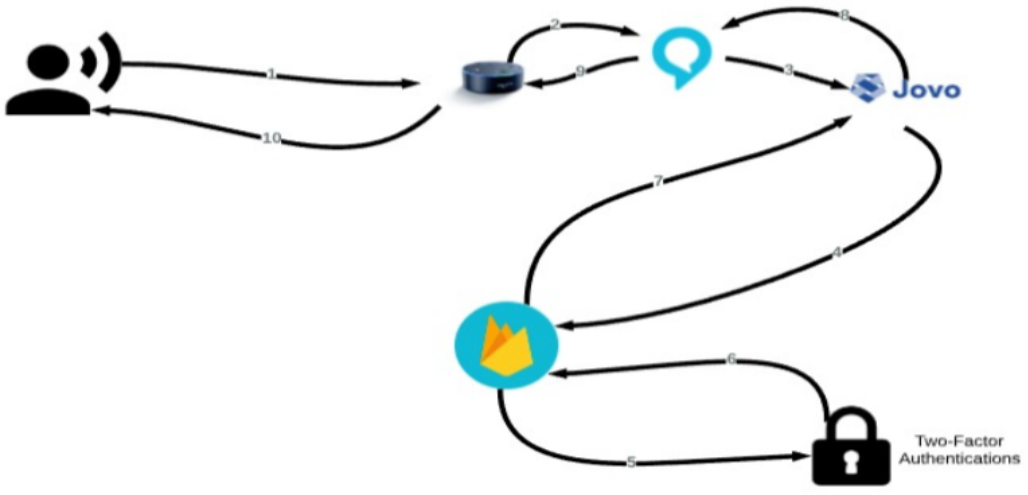


FIGURE 2.2:

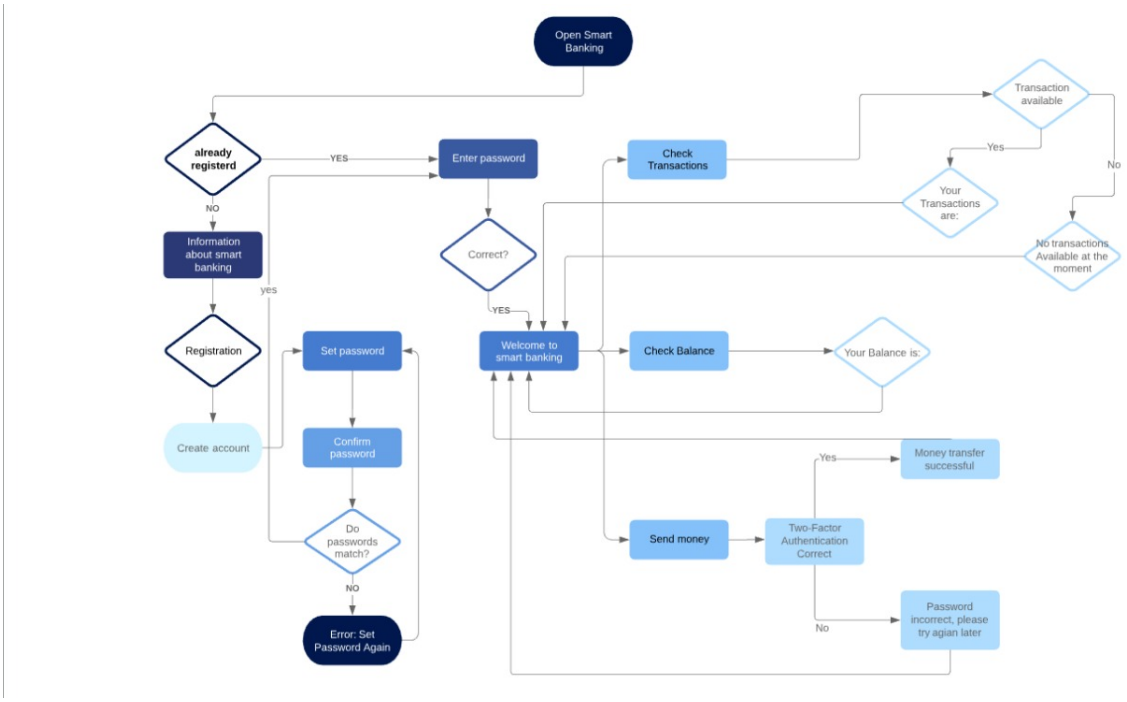


FIGURE 2.3:

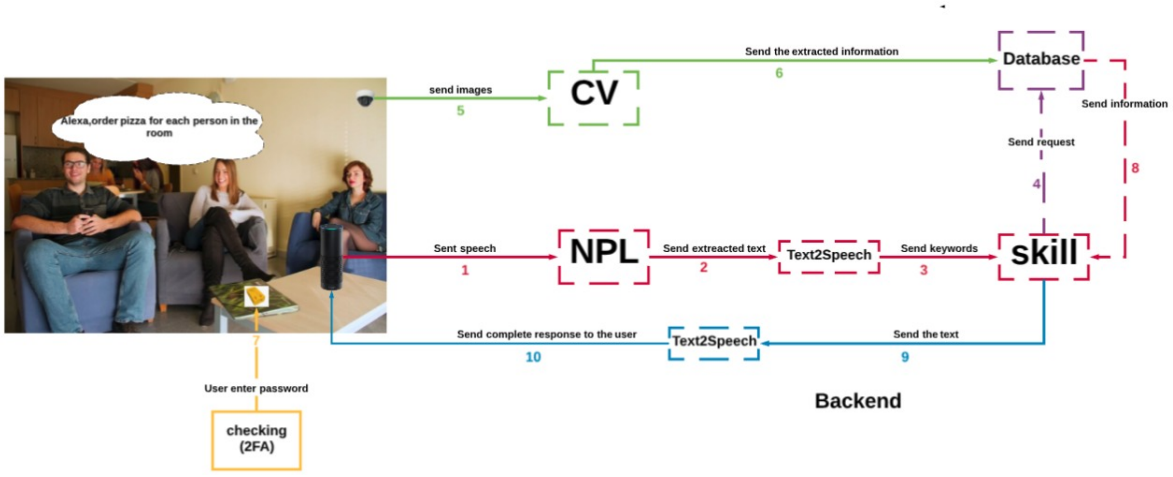


FIGURE 2.4:

2.2.4 System Components

The components that are used in order to implement the Smart banking system are the following:

- Voice Interface
- 3D printed button case

- Alexa Server
- Jovo
- ESP01
- PCB
- Firebase Realtime Database
- Google Cloud Vision API
- GoPro Camera

2.2.5 3D design

In the smart banking system we used a smart button as a layer in security, and in order to make this button usable we had to create a case for it by using Fusion 360 to 3D design it, the design can be shown below:

FIGURE 2.5:
3D Button

2.3 Super Sensor

2.3.1 Problem Description

The development of smart environments and Iot gadgets increased significantly in the previous years.Most houses rely on direct or distributed sensing sensors to measure the surrounding environment,which will required the user to buy different and number of sensors to make their home smarter.This approach limited to the system itself like a smart light switch knows if it is (on/off),plus,it is costly because the user will upgrade their devices and buy smart devices like : light switches, kitchen appliances to control

and monitor all the events in the room. In this project,we developed all in one device that has high capability of sensing all the events in the nearly environment like: The light status,the temperature,the water consumption, etc.This device doesn't required any smart gadgets in the home,so you can monitor your devices using your voice assistance and detect the events using it. We trained our device for many months and different environments,the results shows the ingenuity, accuracy and prospect of this device.

2.3.2 Hypothesis

The system will detect all the event that happened in the room using AI and it is connected to Alexa.

2.3.3 System Architecture

Our objective aims to design a Smart Home for Visually-Challenged People of Determination to help them in their day to day life routines.Currently, the industrial going with the idea of smart homes. Smart home is a house equipped with gadgets that make your life easier like: smart lights, heater, oven and other devices that you control it remotely using your voice or your smart phone. Buying many smart devices to make your home smart is expensive right now as every device alone not cheap. So, we developed super sensor, it is one device that monitor all the activities in your house or office,can be used to gather all sort of information within its perimeter like: the temperature in the room or the number of people and turn on/off a device using your voice assistance .This is by simply placing one device in each room, so we can easily monitor the status of the house.This device should help Visually-Challenged people of determination to interact with the environment around and to know information about it using voice commands.Our system work by:

- First,The user issue a command to any Alexa device.
- Alexa check the Jovo application to open the right skill
- Alexa check the Firebase ,collect the answer then answer the user
- Finnaly ,all the information inside the Firebase is the measurements of the sensors we used ,then using Pyaudio library and machine learning Alexa can answer the status of the room like the temperature ,Water consumption and the light status, etc..

The Architecture of super sensor is shown in Figure 2.6.

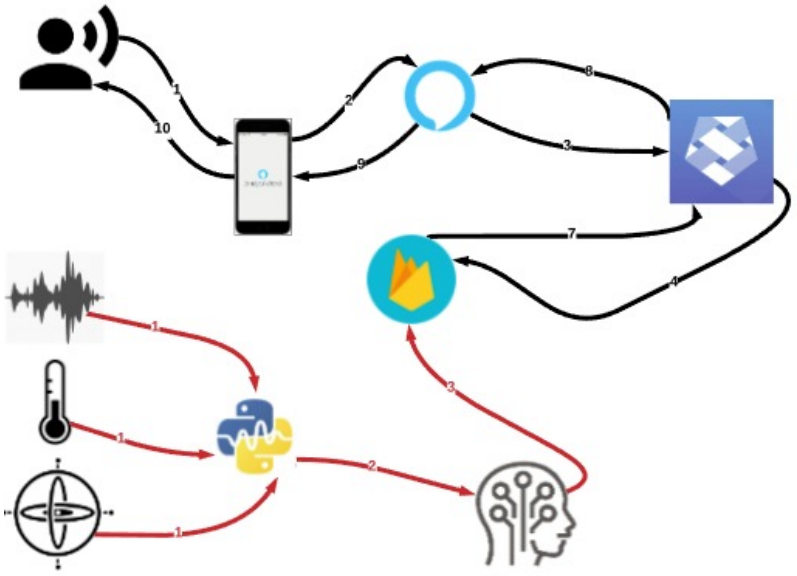


FIGURE 2.6: Super Sensor System Architecture

2.3.4 System Components

The components that are used to implement the Super Sensor system are the following:

- BME280 (Temperature, Humidity and Pressure Sensor)
- LSM9DS1 (Accelerometer, Gyro and Magnetometer)
- TSL2561 (Luminosity Sensor)
- SPH0645 (MICROPHONE BREAKOUT) (Old Implementation)
- Feather M0 (Old Implementation)
- Respeaker Core v2

2.3.5 PyAudio Analysis[1]

Audio data one of the important role in the development of digital content in the market.So the need for techniques that automatically analyze the audio data such as: voice recognition, events recognition, speech analyzes, audio to text analyzes, etc.In this project, we used PyAudio which is an open-source Python library that do real−time digital signal processing and has long list of audio analysis features like:features extraction ,Spectrogram and Chromagram visualization,Beat extraction, Segmentation and Audio−Recording−Functionalities. pyAudioAnalysis is licensed under the Apache License and is available at GitHub[33].We used PyAudioAnalysis to developed device that detect all the events and the smart home functionalities using the Audio Data.The feedback from all these particular audio implementations has guide to functional enhancement of the library.

FIGURE 2.7: PyAudio Analysis

2.3.6 First Design

In our Super sensor we use various number of sensors like BME280, LSM9DS1 ,TSL2561,SPH0645. And Feather M0 as Micro-controller

2.3.6.1 Feather M0

Feather M0 is board developed by Adafruit,it is light and thin and designed to be portable micro−controller.The Feather M0 shipped with the processor ATSAMD21G18 and it clocked at 48 MHz and 3.3v.it comes Micro USB built in so it has USB-to-Serial program & debug capability built This board support different communication chips (WiFi, Bluetooth or radio) to allow for wireless networking.This chip can be connected to 3.7V Lithium polymer batteries for portable project.But,if you don't need batteries the micro USB will run just fine.For more information about the chip pin-out, the Figure 2.8 is provided [34]

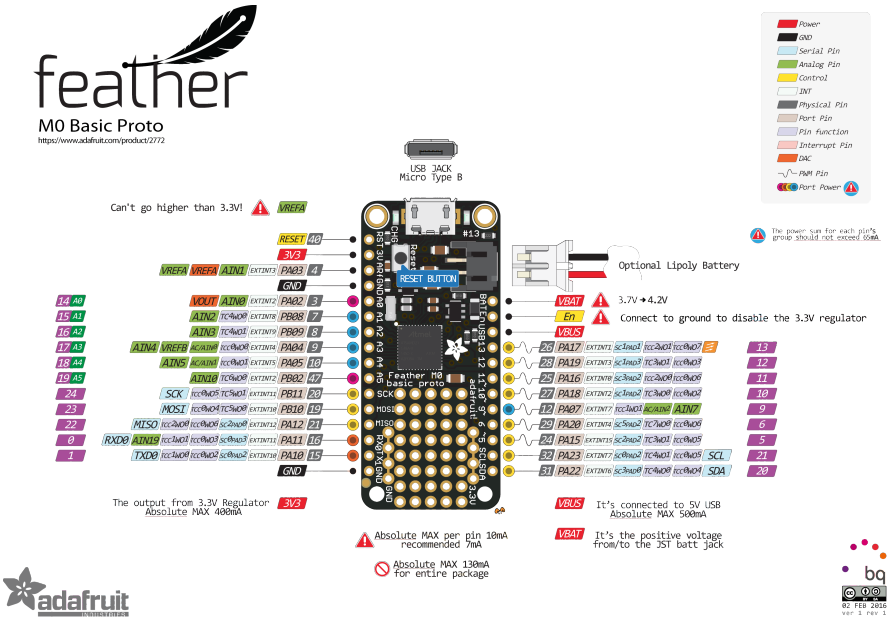


FIGURE 2.8: Feather M0 Pin-out.

2.3.6.2 BME280

The BME280 digital multifunctional sensor developed to measure the humidity with fast response time and high overall accuracy.The pressure sensor in this device give high accuracy and low noise output.This device can also measure the temperature and it is optimized for the lowest noise and highest resolution. for where size and low power consumption are important,and this is the case in our super sensor. It can for measuring temperature, pressure, humidity. The connect to the sensor can be either I2C or SPI.

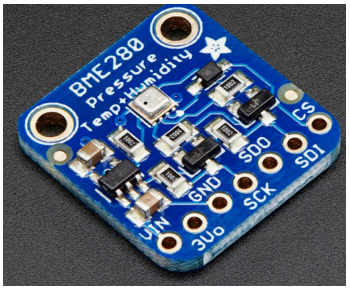


FIGURE 2.9: BME280 Sensor.

2.3.6.3 LSM9DS1

The LSM9DS1 is a classic 9-axis motion sensor module that has 3-axis motion accelerometer, which can measure you direction, orientation, and motion.It include also a 3-axis

magnetometer that can sense the magnetic force. The third is a 3-axis gyroscope that measure or maintain rotational motion.LSM9DS1 can be used with any micro controller and also it offer I2C and SPI connection[35].

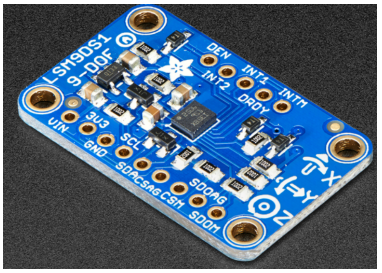


FIGURE 2.10: LSM9DS1 Sensor

2.3.6.4 SPH0645

SPH0645 is purely digital microphone used for small project to recording/detection and can detect sound then convert it to voltage.It has a range of about 50Hz to 15KHz and it use I2S that can take digital audio data in[36].

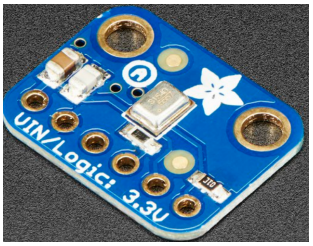


FIGURE 2.11: SPH0645 Sensor

2.3.6.5 TSL2561

The TSL2561 luminosity sensor is a digital light sensor,it is able to detect a wide range of light reading.

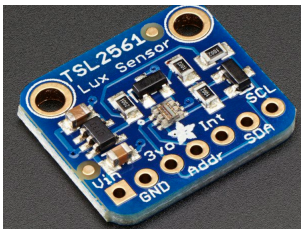


FIGURE 2.12: TSL2561 Sensor

2.3.7 Final Design

We went with the Respeaker in our design,to get more accuracy from the 8 mics around it.Also, The Respeaker can be connected to Alexa which gave as flexibility in the smart homes department.Finally the low computing power of the Respeaker.The Figure 2.13

FIGURE 2.13: The Final Design For Super Sensor

2.4 Circuit Design

2.4.1 Circuit Design For Smart Banking Key chain (2FA)

We used Fritzing to design our circuit and PCB because it is easier than Eagle and the way to add libraries is simple. Initially ,we started our research on finding a small device that is portable and doesn't consume power and we chose ESP01, Additionally, we needed a button for 2 factor Authentication,another one for restart and LED. The following were the features of the rst circuit.

- ESP01.
- One button For One time password (OTP).
- One Led to represent the status of the transaction.
- One button for restart.
- Pin Out for battery.

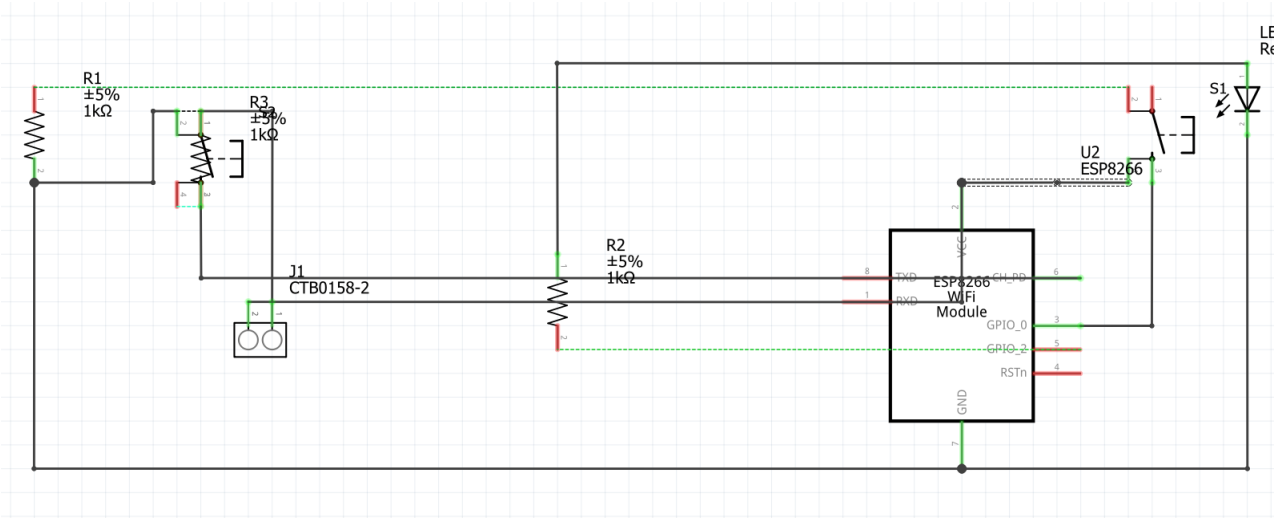


FIGURE 2.14: Circuit Diagram For Smart Banking Key chain (2FA)

2.4.2 Circuit Design For Super Sensor

We tried designed a circuit that include all the sensors in one PCB.The following were the features of the rst circuit.

- Adafruit TSL2561
- Adafruit BME280
- Adafruit LSM9DS1
- Pin Out for battery

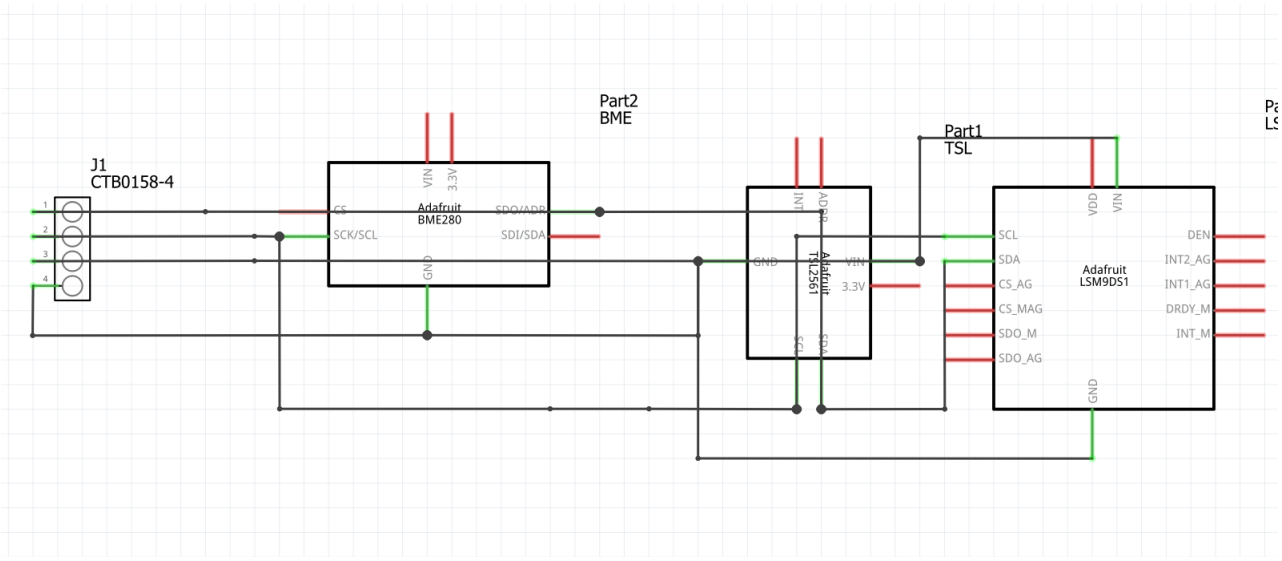


FIGURE 2.15: Circuit Diagram For Super Sensor

2.5 PCB Design

The purpose of designing PCB is to have small form factor that contain all the important components of the design.

2.5.1 PCB For Smart Banking Key chain (2FA)

For the Final PCB we removed the second button for restart the device and we hard coded the button when we press long press to restart as shown in Figure 2.17.

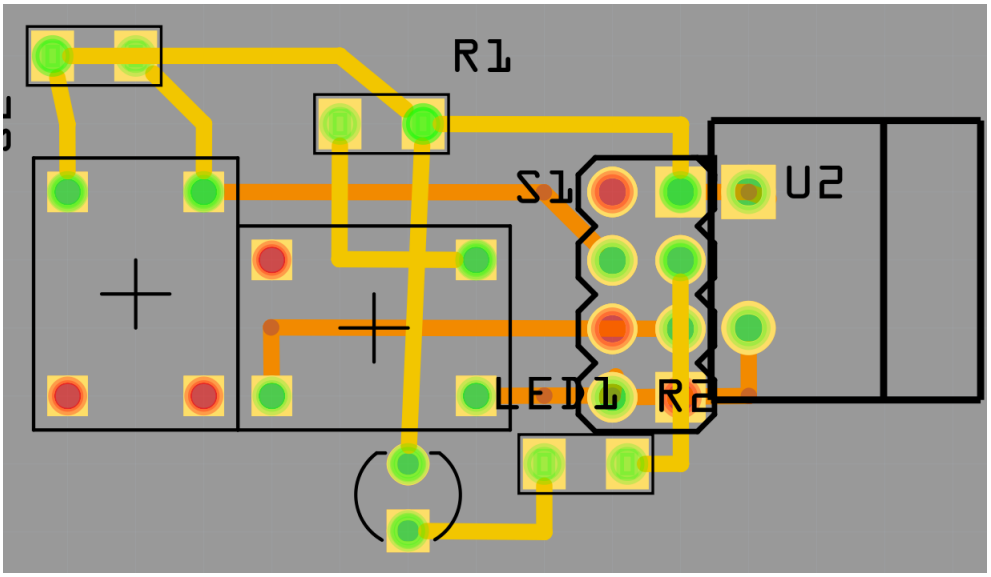


FIGURE 2.16: PCB For Key chain

FIGURE 2.17: PCB For Key chain in Real Life

2.5.2 PCB For Super Sensor

We compacted all the sensors in one PCB ,then we solder it in small form factor as shown in Figure 2.19

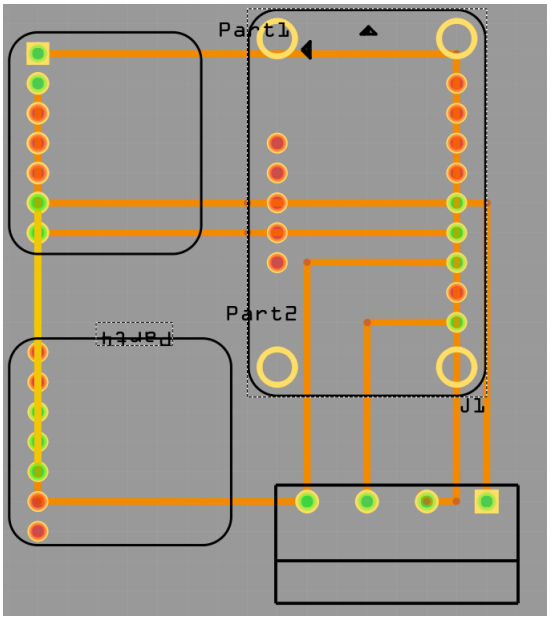


FIGURE 2.18: PCB For Super Sensor

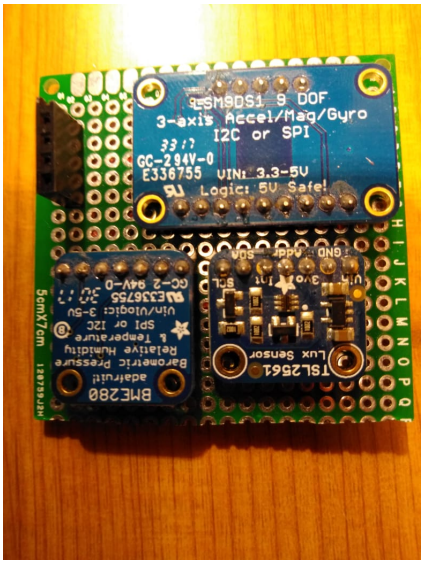


FIGURE 2.19: PCB For Super Sensor in Real life

Chapter 3

Alexa Voice Service

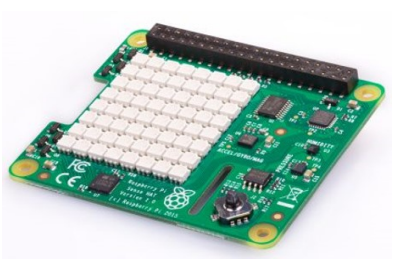
3.1 Respeaker

In this project, we had to use some type of single board computer which contains a SOC (System On Chip) in order to have our VUI (Voice User Interface) system. Initially, we chose Amazon Alexa Echo, but it has no pins so we could not attach our Camera. Figure 3.1 shows a picture of Alexa Echo.

FIGURE 3.1: Alexa Echo

Our second choice was a board that we are very familiar with; the Raspberry Pi 3 Model B+. We also planned to use the Sense Hat along with the Pi because it has almost everything that we need except the microphones. So, we got the microphones separately from the Hat but we faced a problem with the placing of the microphone which we could have solved somehow with creating a 3D-design of a case that can cover the Pi, Hat and the microphone without having any sort of blocking in front of the microphone. But, we had a second problem which was that we ran out of pins, because the Sense Hat takes almost all the pins of the Pi leaving no thing to microphones, which need to be a lot and in every direction for our application. After some thinking, we have decided to replace the whole single board computer and look for a more specified application single board computer. Figure 3.2 shows the Raspberry Pi, Sense Hat and the Microphone.

(a) Raspberry Pi



(b) Sense Hat

(c) The Microphone

FIGURE 3.2: Our VUI Components

Our third choice was the MATRIX Creator, a single board computer which consists of 8-Mic Array. The MATRIX Creator was our best option but due to problems with the vendors we were not able to get our hands on the board. Figure 3.3 shows the MATRIX Creator.

FIGURE 3.3: MATRIX Creator

Our final choice a single board computer which we used to implement our Voice User Interface was the ReSpeaker Core v2.0. We choose the ReSpeaker because it offered everything that the previous boards have lacked. More details about the ReSpeaker can be found in the Appendix section. Figure 3.4 shows the ReSpeaker.

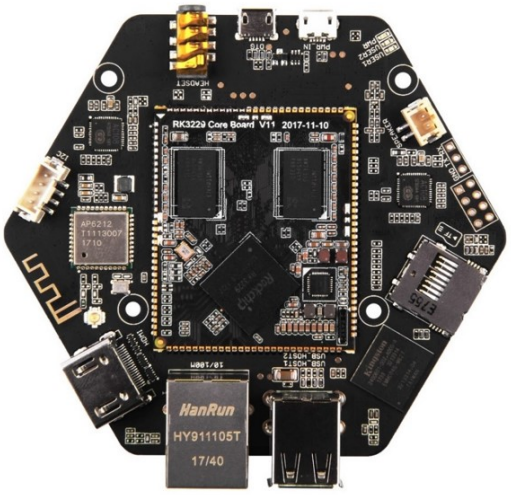


FIGURE 3.4: The ReSpeaker

3.2 Respeaker Setup [2]

3.2.1 Image Installation

Before we start we need to Setup the Respeaker.First,we need to burn the ReSpeaker Core v2.0 image and install it on SD card and then boot from it.

1. Download one of the images from the cloud [37],either respeaker-debian-9-lxqt-sd-********-4gb.img.xz or respeaker-debian-9-iot-sd-********-4gb.img.xz. We went with lxqt version because it comes with a desktop GUI while the iot version does not.
2. Using SD card reader plug the SD to the computer.The minimum capacity for an SD card should be more than 4G .
3. Unzip the file you downloaded from step one a *.img file,then burn it using any writing tools like Rufus to your SD card,the burn will take about 10 minutes to complete.
4. Insert the SD card after the burning process into the ReSpeaker.
5. Power the board using the PWR-IN micro usb port, the SD card should be insert and Do Not remove it until the user1 and user2 stop blinking.User1 LED blink in a heartbeat pattern and user2 LED usually light during SD card accesses.

3.2.2 Serial Console

To access the ReSpeaker there is three ways: using HDMI cable, Putty or VNC Viewer.

3.2.2.1 HDMI Cable

We need to configure Alexa using the internet ,so, the OTG connection is not enough ,in this case, we need to use the HDMI port to a HDMI or HDMI to VGA converter as wee can see in Figure 3.5.To use the ReSpeaker we will use monitor,mouse and keyboard in order to use the interface

3.2.2.2 Putty

To access the ReSpeaker Using Putty we need to follow these steps:

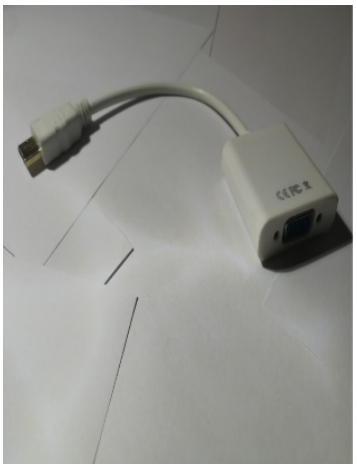


FIGURE 3.5: HDMI to VGA Converter

1. There is two micro USB ports on the ReSpeaker,one for PWR-In which support only powering the device and OTG for send and receive.Using micro USB cable (it should support send data and power), plug the cable to the ReSpeaker's OTG micro USB port and then to the PC.
2. Check the device manger and look for new serial deviced named COMx (x could be any number)
3. Open PUTTY, select Serial protocol, enter the COMx port of the ReSpeaker fill in the correct COM port of the ReSpeaker and 115200 baud.
4. The default user name and the password is respeaker.

3.2.2.3 VNC

We found that the best solution is to use VNC because it can connect to the ReSpeaker from anywhere just by entering the ReSpeaker IP, and it allow us to work on the computer browser.Also, the ReSpeaker come with pre-installed VNC server(X11VNC server) as we can see in Figure 3.6.

To access the ReSpeaker Using VNC Viewer we need to apply the Putty steps first then we follow these steps:

1. Configure the ReSpeaker's network using the Network Manager tool using nmtui. nmtui will be pre-installed on the ReSpeaker image,to use nmtui we write this command in Putty:

```
sudo nmtui
```

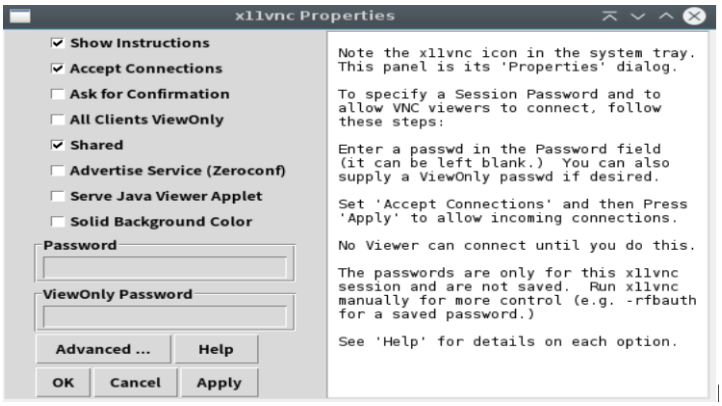


FIGURE 3.6: X11VNC Server

2. After that a configuration page will appear press on Activate a connection.The Figure 3.7 show the configuration window that will appear.

FIGURE 3.7: Wifi Configuration Page

3. Search for your Wi-Fi network then press Enter, type the Wi-Fi password then press Enter. The * mark in the Figure 3.8 means that you are connect now to the Wifi network To leave the network manger you need to tap Esc key twice.

4. To find the IP address of the ReSpeaker we use the command below.

```
ip address
```

The IP address for the ReSpeaker is 192.168.0.46(The address will be different) as we see in Figure 3.9

5. Next,we changed the ReSpeaker IP from dynamic IP to static IP by entering This command :

```
sudo nano /etc/dhcpcd.conf
```

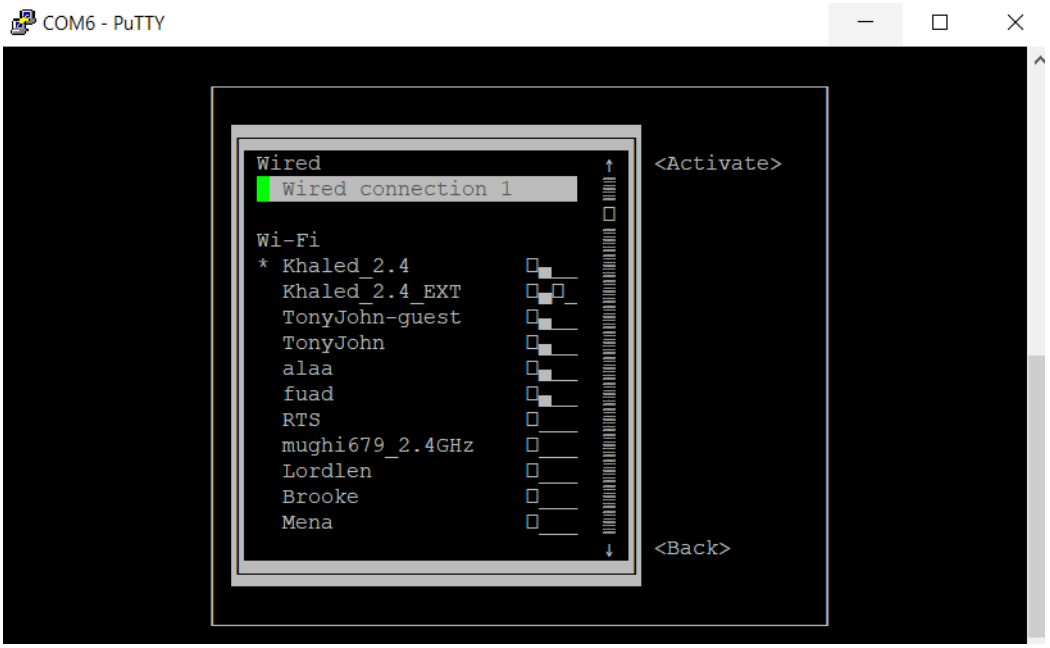


FIGURE 3.8: Wifi Configuration Successfully

```
interface eth0 static ip_adress=192.168.0.46/24
static routers=192.168.1.1
static domain_name_servers=192.168.1.1
```

6. But when we tried to connect we got this error 3.10

7. We tried to use SSH we got this error 3.11

8. The client was not able to find the server IP address,so we tried to ping the server from the client but we keep getting timeout error as Figure 3.12 represent.

9. The solution was that you must ping the client from the server.As we can see in Figure 3.13 it works.

   So, now we can connect easier next time.

10. To use VNC, the Respeaker and the PC should be connoted to the same network.Then open VNC Viewer, type the ReSpeaker ip address at the address bar as Figure 3.14 .Warning window will pop-up that connection is Unencrypted, click Connet to go on. The password is Respeaker.

11. After each restart the was issue that we should ping first the client from the server before connecting which is not practical,so ,after some research about the issue we found out that it caused by the power management system in the ReSpeaker which turning off the NICs to conserve power.We don't want to disable the power

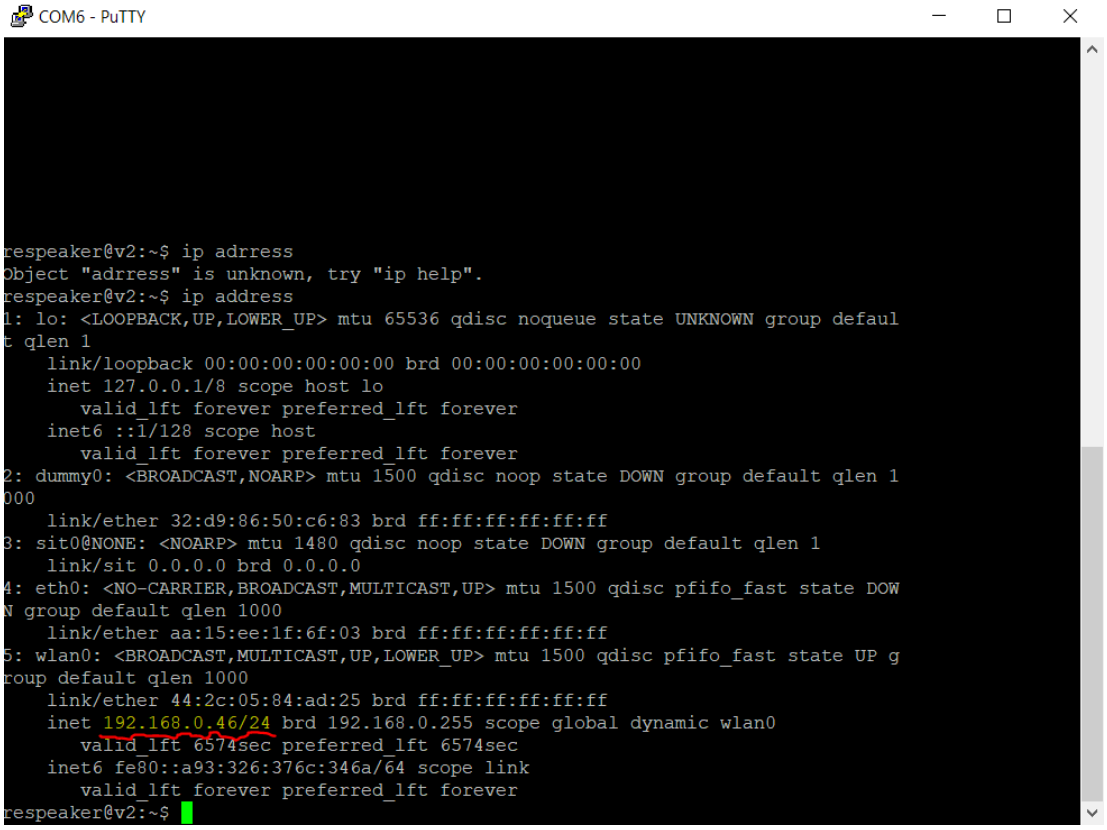


FIGURE 3.9: Respeaker IP Address

management system. so, the best solution was to scan the whole network on start-up.First ,we need to install nmap which is a free and open-source security scanner.Then create a script called nmap.sh and edit script by writing :

```
sudo nmap -sP 192.168.1.0/24
```

Also, we need to edit the rc.local file that execute on start-up by writing:

```
/home/respeaker/Desktop/nmap.sh || exit 1
```

12. On the next start-up everything should work fine

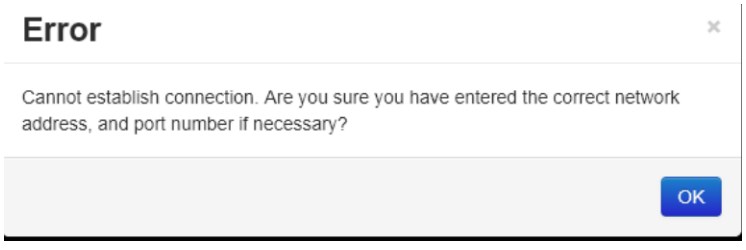


FIGURE 3.10: Error in VNC

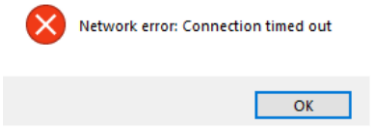


FIGURE 3.11: SSH Error

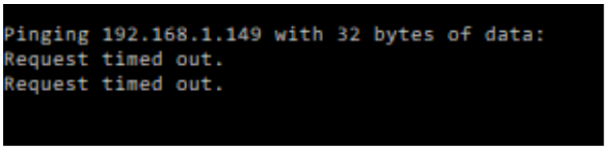


FIGURE 3.12: Timeout Error

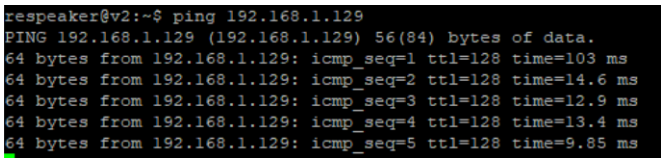


FIGURE 3.13: Ping The Client

3.2.3 Alexa Setup

In this part we will do the setup for Alexa and will be introduced to librespeaker. The librespeaker is an audio processing library which can perform:

- Noise suppression
- Direction of arrival calculation
- Beamforming
- Hotword searching
- Acoustic echo cancellation

librespeaker work by reading the mic stream from Linux sound server.Then detect a few APIs which allow users to get indicated when specific is said and the processed mic data in PCM format,after that, it can be sent to cloud services for further processing

1. We need to download the packages by entering this command in the terminal:

```
curl https://raw.githubusercontent.com/respeaker/respeakerd/master/scripts/inst
all_all.sh|bash
```

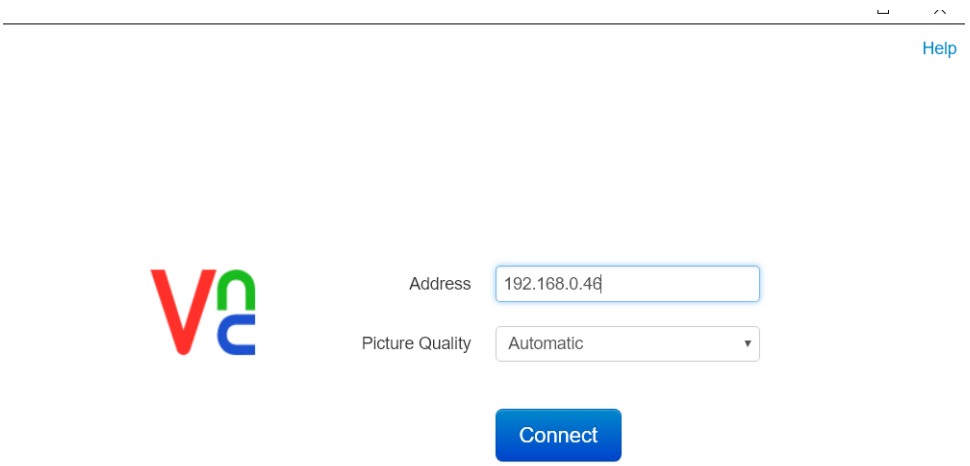


FIGURE 3.14: VNC IP Setup

2. We need to authorize Alexa, this can done by connect to the ReSpeaker using VNC. Then open the Internet Browser, and tap 127.0.0.1:3000 at the URL input field .This will forward us to amazon login page as we can see in this Figure 3.15,so we sign in using our account

3. After finishing the steps above we Run the demo to test Alexa by typing:

```
python /home/respeaker/respeakerd/clients/Python/demo_respeaker_v2_vep_alexa_with_light.py
```

4. Say wake up Alexa , or Alexa and it should answer you.

3.2.4 Play with AVS

3.2.4.1 C++

To run amazon official AVS C++ SDK with respeaker,you should have technical background about Linux OS.This guide will show you how.

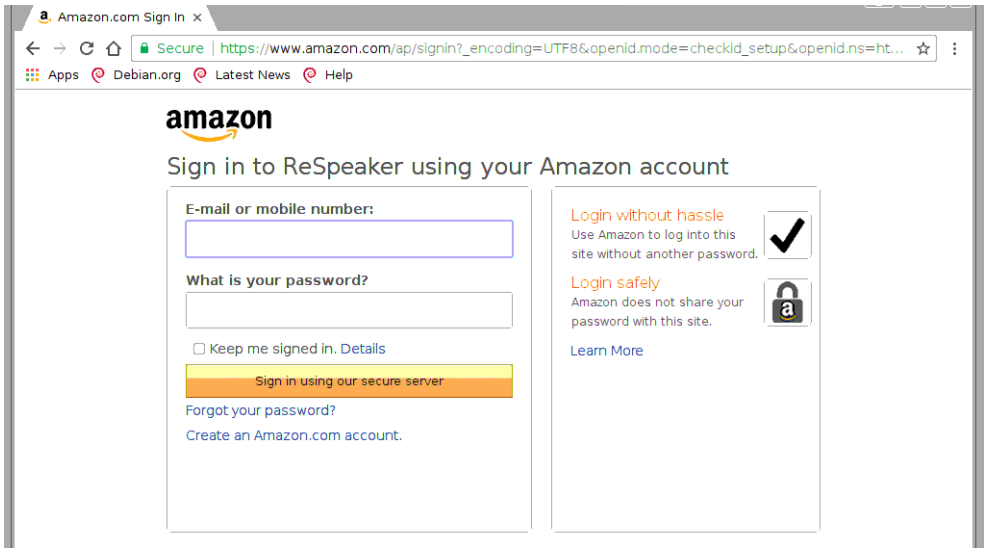


FIGURE 3.15: Alexa Authorize Page

3.2.4.2 Preparation

To install the basic software packages run the command below in terminal: erminal:

```
## install deps
sudo apt update
sudo apt install -y librespeaker git cmake
sudo apt install -y python-mraa python-upm libmraa1 libupm1 mraa-tools
sudo pip install pixel_ring pydbus

cd /home/respeaker
git clone https://github.com/respeaker/respeakerd.git

cd /home/respeaker/respeakerd

sudo cp -f build/respeakerd /usr/local/bin
sudo cp -f scripts/respeakerd_safe /usr/local/bin
sudo chmod a+x /usr/local/bin/respeakerd
sudo chmod a+x /usr/local/bin/respeakerd_safe
sudo mkdir -p /usr/local/etc/respeakerd
sudo cp -Rf build/resources /usr/local/etc/respeakerd/
sudo cp -f scripts/respeakerd.service /etc/systemd/system/


#enable system service
sudo systemctl enable respeakerd
sudo systemctl start respeakerd
```

3.2.4.3 PulseAudio Configuratin

- Use nano to edit default.pa.Please type the following command to edit:

```
sudo nano /etc/pulse/default.pa
```

- Once the file is open, press I to edit.Then copy and paste the lines below at the end of this file:

```
load-module module-pipe-source source_name="respeakerd_output" rate=16000 channels=1
set-default-source respeakerd_output
```

- Press Esc to exit the nano command.
- Press : to access the command mode, then tap W then Enter to save.
- After saving ,press Q then Enter to exit.
- The end of the default.pa file should look like Figure 3.16
- Reboot the board by entering :

```
sudo reboot -f
```

```
### Cork music/video streams when a phone stream is active
load-module module-role-cork

### Modules to allow autoloading of filters (such as echo cancellation)
### on demand. module-filter-heuristics tries to determine what filters
### make sense, and module-filter-apply does the heavy-lifting of
### loading modules and rerouting streams.
load-module module-filter-heuristics
load-module module-filter-apply

### Make some devices default
#set-default-sink output
#set-default-source input
load-module module-pipe-source source_name="respeakerd_output" rate=16000 channels=1
set-default-source respeakerd_output
```

FIGURE 3.16: default.pa File Configuration

3.2.4.4 Start PulseAudio mode

The respeakerd works in PulseAudio mode by simply output the processed audio stream which is created by the module-pipe-source of PusleAudio to a named pipe

- To start the PulseAudio mode enter the commands below:

```
sudo systemctl stop respeakerd
```

- Edit the respeakerdsafe file by entering the command below:

```
sudo vim /usr/local/bin/respeakerd_safe
```

- Modify the file as the following code:

```
#!/bin/bash

pulseaudio --check

while [ $? == 1 ]; do
    sleep 1
    pulseaudio --check
done

while [ ! -p /tmp/music.input ]; do
   sleep 1
done

sleep 5

/usr/local/bin/respeakerd
--Alexa_res_path="/usr/local/etc/respeakerd/resources/common.res"
--Alexa_model_path="/usr/local/etc/respeakerd/resources/Alexa.umdl"
--Alexa_sensitivity="0.4" --source="alsa_input.platform-sound_0.seeed-8ch" --mode=pulse
```

- Restart the service by enter:

```
sudo systemctl start respeakerd
```

3.2.4.5 Compile and Run AVS C++ SDK

- First,Download and install the necessary files by these command:

```
    cd /home/respeaker/ && mkdir sdk-folder && cd sdk-folder &&
    mkdir sdk-build
    sdk-source third-party application-necessities && cd
    application-necessities && mkdir sound-files
sudo apt-get -y install git gcc cmake build-essential libsqlite3-dev
libcurl4-openssl-dev libfaad-dev libsoup2.4-dev libgcrypt20-dev
libgstreamer-plugins-bad1.0-dev gstreamer1.0-plugins-good libasound2-dev
doxygen
cd /home/respeaker/sdk-folder/third-party && wget -c
http://www.portaudio.com/archives/pa_stable_v190600_20161030.tgz && tar zxf
```

```
pa_stable_v190600_20161030.tgz && cd portaudio && ./configure --without-jack &&
make
sudo pip install commentjson
sudo pip install flask
sudo pip install requests
cd /home/respeaker/sdk-folder/sdk-source && git clone
git://github.com/respeaker/avs-device-sdk.git
cd /home/respeaker/sdk-folder/sdk-build && cmake
/home/respeaker/sdk-folder/sdk-source/avs-device-sdk -DCMAKE_BUILD_TYPE=DEBUG
-DRESPEAKERD_KEY_WORD_DETECTOR=ON -DGSTREAMER_MEDIA_PLAYER=ON -DPORTAUDIO=ON
-DPORTAUDIO_LIB_PATH=/home/respeaker/sdk-folder/third-party/portaudio/lib/.libs
/libportaudio.a -DPORTAUDIO_INCLUDE_DIR=/home/respeaker/sdk-folder/third-party/
portaudio/include
make SampleApp -j2
```

3.2.4.6 Get Authorization of AVS

This section will guide you to setup and run a local authorization server, which we'll use to obtain a refresh token. This refresh token used for exchange access token along with Client ID and Client Secert, it is a need for the sample app to send to Alexa with each event (request).

- Register your product with Amazon using these instructions[38] to register your product and create a security profile. Skip this step if you already have a registered product.Make you know the Product ID from the Product information page, also the Client ID and Client Secret from the Security Profile page. You'll this to configure the authorization server.
- Update AlexaClientSDKConfig.json by the command below:

```
nano /home/respeaker/sdk-folder/sdk-build/Integration/AlexaClientSDKConfig.json
```

Replace the contents of AlexaClientSDKConfig.json with this JSON :

```
{
    "authDelegate":{
        "clientSecret":"YOUR_CLIENT_SECRET",
        "deviceSerialNumber":"123456",
        "refreshToken":"",
        "clientId":"YOUR_CLIENT_ID",
        "productId":"YOUR_PRODUCT_ID"
    },
    "alertsCapabilityAgent":{
        "databaseFilePath":"/home/respeaker/sdk-folder/application-necessities/alerts.db"
    },
    "settings":{
        "databaseFilePath":"/home/respeaker/sdk-folder/application-necessities/settings.db"
        "defaultAVSClientSettings":{
```

```
            "locale":"en-US"
        }
    },
    "certifiedSender":{
        "databaseFilePath":"/home/respeaker/sdk-folder/application-necessities/
        certifiedSender.db"
    },
    "notifications":{
        "databaseFilePath":"/home/respeaker/sdk-folder/application-necessities/n
        otifications.db"
    }
}
```

Enter the clientId, clientSecret, and productId that you already have during device registration and save.

- Obtain a refresh token by open the browser and navigate to 127.0.0.1:3000. Login using your Amazon credentials and follow the instructions provided.
- After succeed you will see This message 3.17

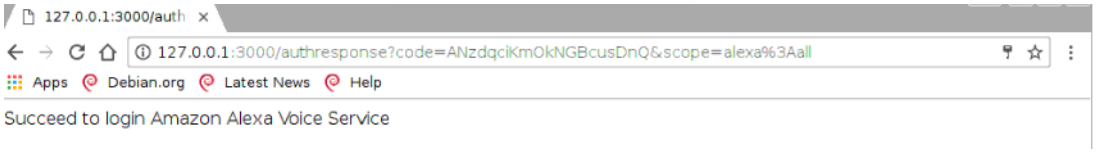


FIGURE 3.17: Amazon Authorize succeed

- Write the command below to test the AVS configuration:

```
/home/respeaker/sdk-folder/sdk-build/SampleApp/src/SampleApp
/home/respeaker/sdk-folder/sdk-build/Integration/AlexaClientSDKConfig.json
```

- A window for Sample APP will appear as we can see in Figure 3.18.Now you are able to make conversations with Alexa using command line messages.
- To activate the on-board LED effect,enter the commands below:

```
sudo cp -f /home/respeaker/respeakerd/scripts/pixel_ring_server /usr/local/bin/
sudo chmod a+x /usr/local/bin/pixel_ring_server
pixel_ring_server
```

- To wake up the ReSpeaker by a special word.enter the commands below:

```
sudo cp -f /home/respeaker/respeakerd/scripts/avs_cpp_sdk_safe /usr/local/bin
sudo chmod a+x /usr/local/bin/avs_cpp_sdk_safe
sudo cp -f /home/respeaker/respeakerd/scripts/pixel_ring_server.service
/etc/systemd/system/
sudo cp -f /home/respeaker/respeakerd/scripts/avs_cpp_sdk.service /etc/systemd/system/
sudo systemctl enable pixel_ring_server
sudo systemctl enable avs_cpp_sdk
```

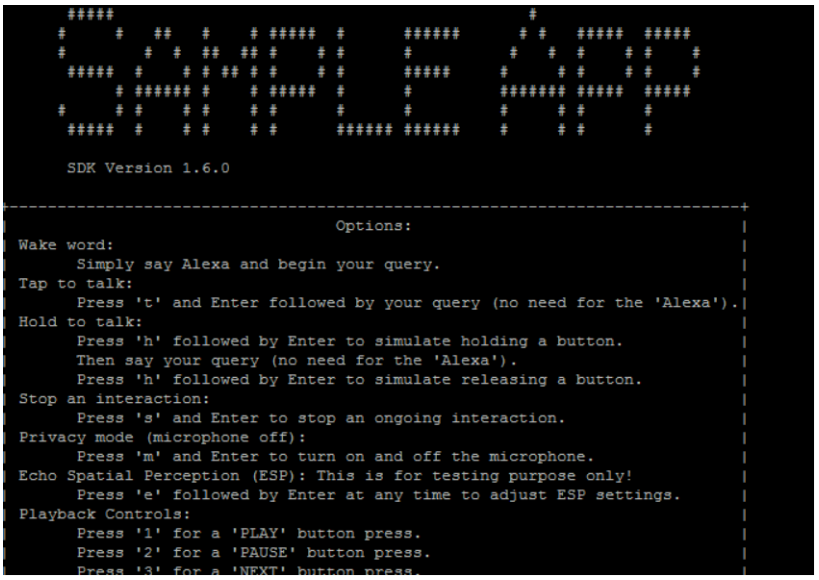


FIGURE 3.18: Sample App Window

```
sudo systemctl start pixel_ring_server
sudo systemctl start avs_cpp_sdk
```

- Now call Alexa and she will answer back.

3.3 Jovo Setup[3]

There are a few stages that happen. The three stages that happen is:

- A user talking to any Alexa device
- The Alexa API create a request based on the understands what the user wants
- The Skill code knows what to do when it receive the request.
- Then this circle go backward.

The architecture of Alexa and how users interact with its Skills shown with details in Figure 3.19.

First wee need to configure the skill in order to work.We will use Amazon Developer Portal to configure it.

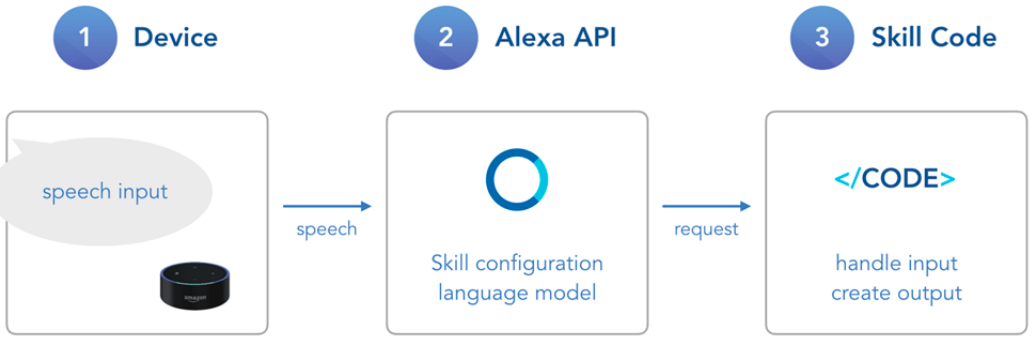


FIGURE 3.19: The Architecture of Jovo

3.3.1 Create a Skill using Amazon Developer Portal

- Visit developer.amazon.com then click "Developer Console" on upper corner of the page as shown in Figure 3.20

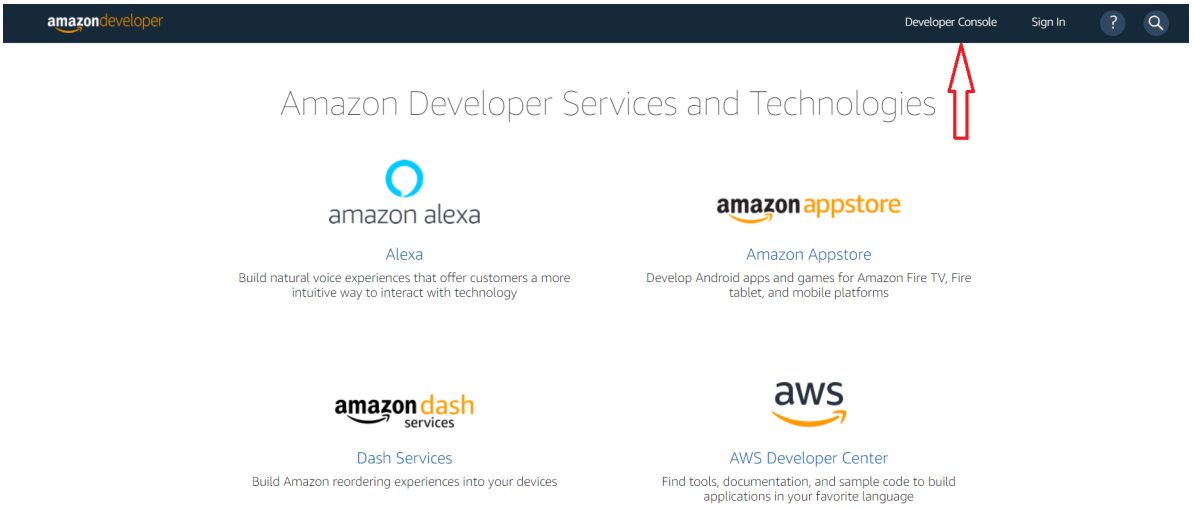


FIGURE 3.20: Amazon Developer Portal

- Login Page will apear as shwon in Figure 3.21. If you new user create new account, if you old user signin using your account.
- To create a new skill for your project, click on the "Alexa" menu in the bar up and click on "Alexa Skill Kit" as shown in Figure 3.22
- A page will apear click on the blue button that named "Create Skill" as shown in Figure 3.23
- Next, name your Skill.

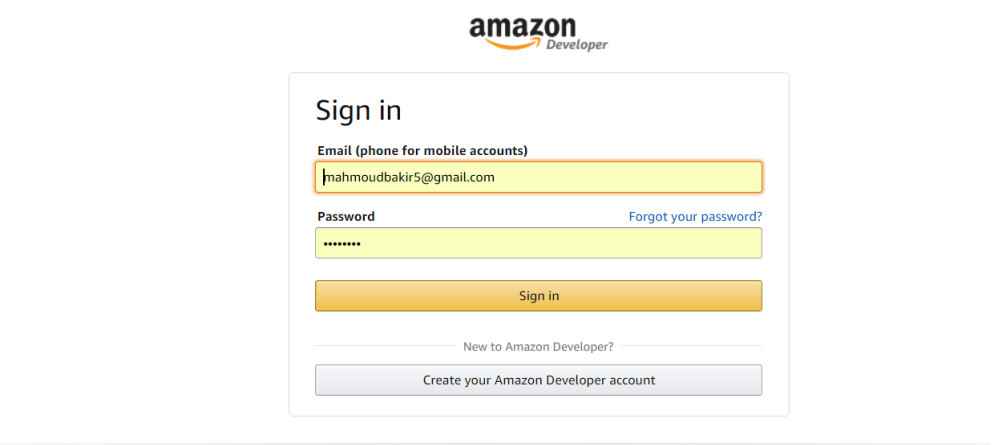


FIGURE 3.21: Amazon Developer Portal Signin Page

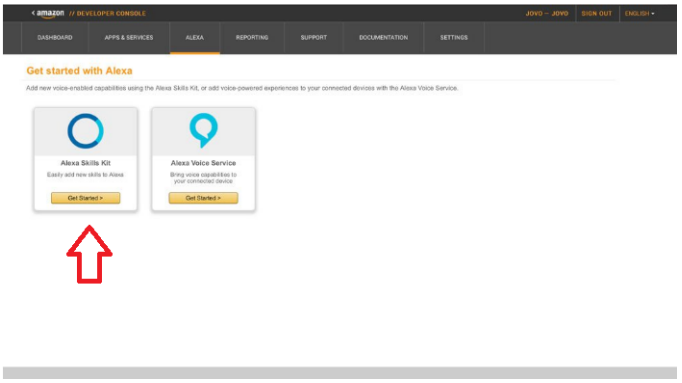


FIGURE 3.22: Alexa Skill Kit

- Choose "Custom" configuration for the Skill model.
- the Skill home page should look like Figure 3.24

3.3.2 Create a Language Model

Alexa works by it the user commands (speech or text) and transforms it into text.Then, it uses a language model to understand what the user ask for ((natural language understanding) Alexa made of four elements:

- Invocation,
- Intents
- Utterances
- Slots.

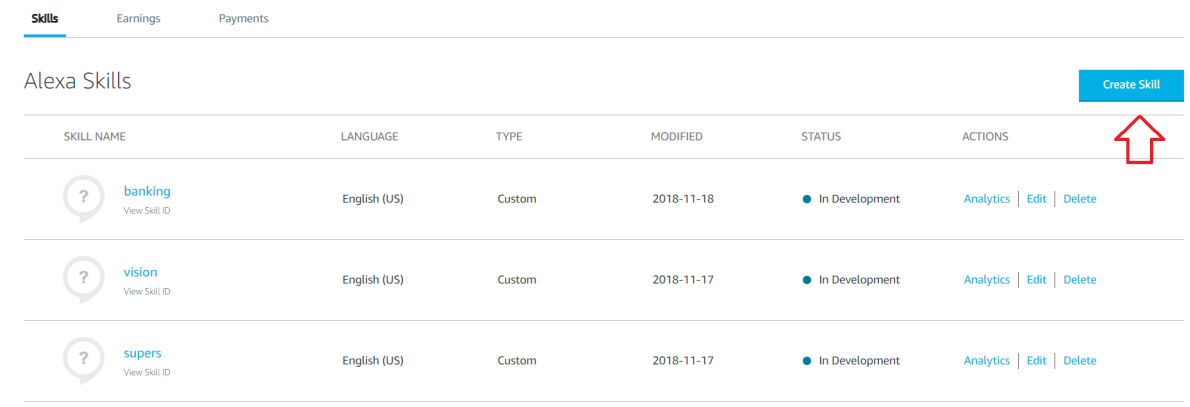


FIGURE 3.23: Create Skill

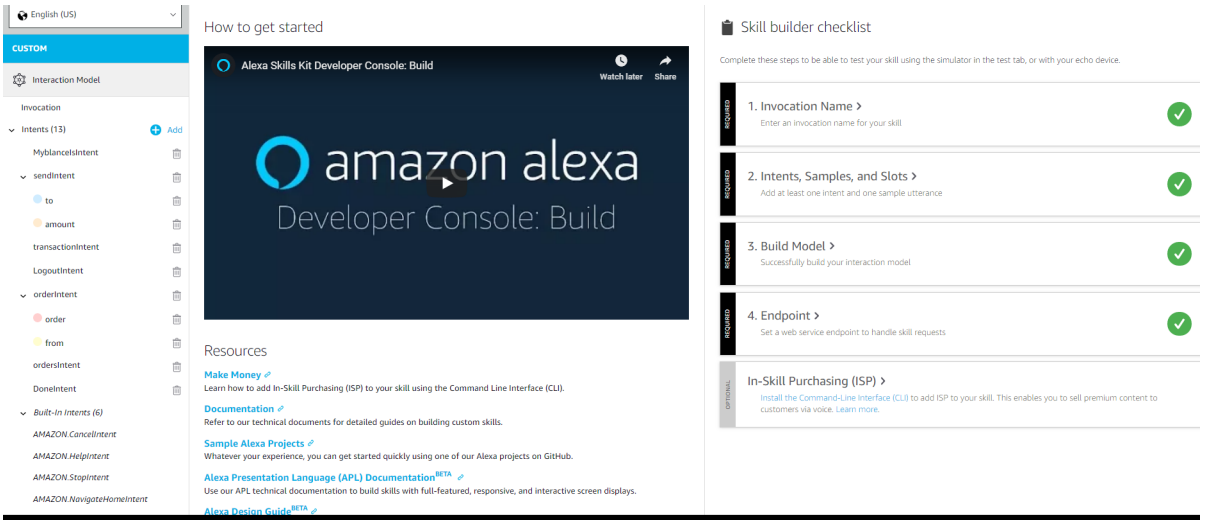


FIGURE 3.24: Skill Home Page

3.3.2.1 Invocation

Alexa Skill has two types of names: The name which is the one that people see in their Alexa app and the invocation name which is the one used to access the Skill.Figure 3.25 explain the idea of invocation

3.3.2.2 Intents

The intent thing the use wanted while talking to device.It is the basic meaning that can be extracted from the sentence that user is asking for.For end up with that specific intent there is several ways.Like when you ask for FindRestaurantIntent it could have

FIGURE 3.25: Alexa Skill Name Types

different direction as shown in Figure 3.26,because the users could express it in many ways.So Alexa language models use utterances.

FIGURE 3.26: Intents for FindRestaurantIntent

3.3.3 Utterance

An utterance is the sentence the user is saying .But there is a large An utterance (sometimes called user expression) is the actual sentence a user is saying. There is huge set of utterances that could be appropriate into the same intent. So Alexa uses slots.

3.3.4 Slots

The slots is the specific information that the user gave to the device in order to have the best experience in understanding(serve the user intent). Maybe the user want a cheap place to eat,find a restaurants near to me or restaurants that serve sushi.The Figure 3.27 demonstrate the idea of slots.

For each Skill there will be different setup.We will discuss the setup for each Skill in their Implementation 4.

FIGURE 3.27: Slots

Chapter 4

Implementation

4.1 Vision

4.1.1 Web Application:Angular

After we finished from implementing the Jovo skill,we require a comprehensive platform that we can use to create a spot or a a room and then add the desired description. We did some research and came out with the Angular platform (Version 6) as it was a best fit to our case, so we implemented our Web Application using Angular. The following steps shows the Implementation process:-

1. To start, we need first to install the Angular Command Line CLI using the following command:-

```
npm install -g @angular/cli
```

   This will start installing all the important packages and libraries needed by Angular while also including the functionality that Webpack offers.
   But, before installing we have to make sure that Node and Npm are both installed and up to date in our system as they both are necessary to Angular.

```
node -version
npm -version
```

   In our case, luckily everything was up to date Node (V8.12.0) and Npm (V6.4.1).

2. After installing the Angular CLI successfully, we can now create our new Angular project using this command:

```
ng new Vison
```

This command will install all the necessary dependencies and libraries needed for the Angular project.

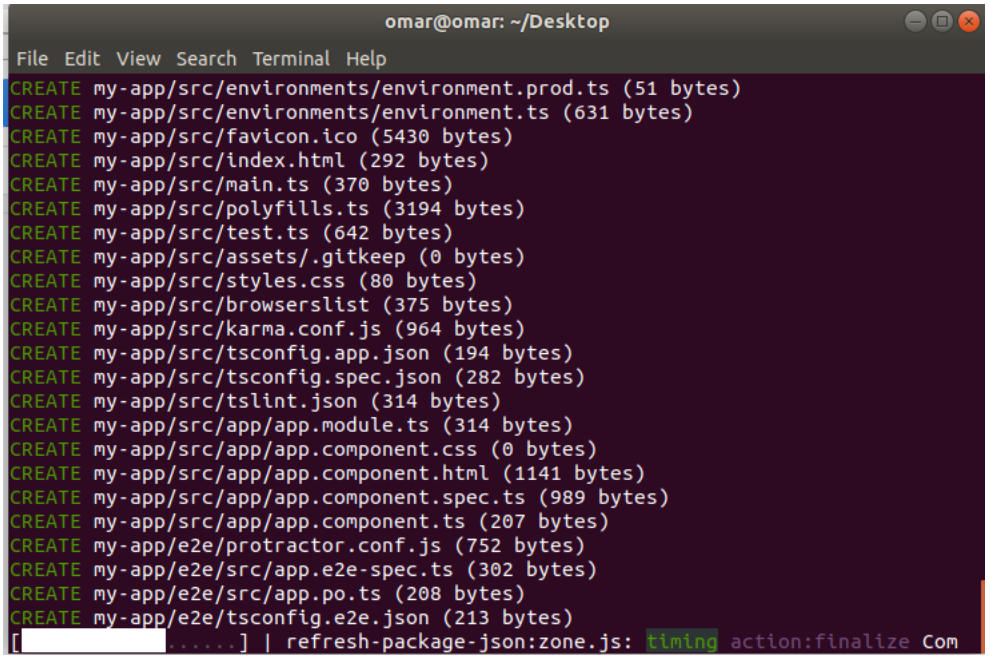


FIGURE 4.1: ng new

3. Now if we run the command

```
Npm start
```

The Angular project will compiled and the project will be hosted on port 4200.

4. Now we can start actually working on our Angular project.First of all we need to add a SideBar to be able to move between various pages. In our case we will have two pages where the first page is used to display all available rooms and the second page is used for editing and adding more rooms.
For that we start by removing everything inside the file App.ts in order to get rid of the default page loaded by Angular,Then we will create a SideBar component using the following command:

```
ng generate component sidebar
```

This command will generate the TS and SCSS and HTML files for the SideBar component also, it will automatically import the component in the app.module file

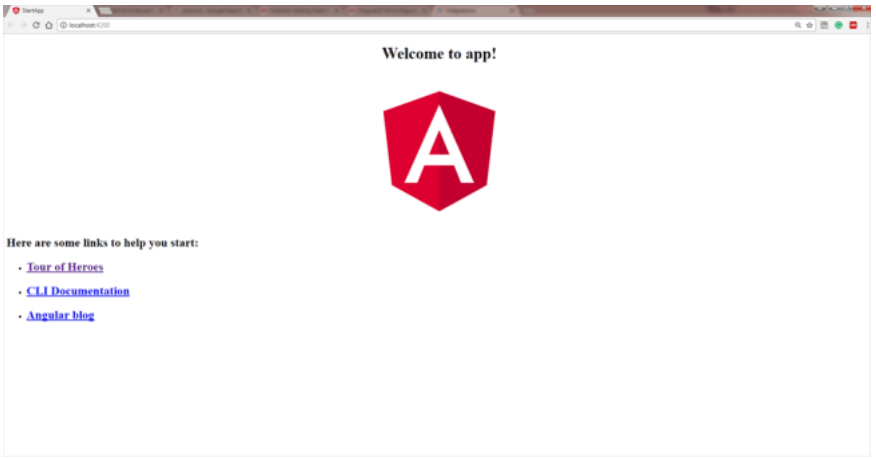


FIGURE 4.2: port 4200

which is responsible for handling all the libraries and devices used in the Application.Then, we will write all of our logic inside the TS file while having the HTML file being responsible for creating the content and finally the SCSS file which is used for style format.

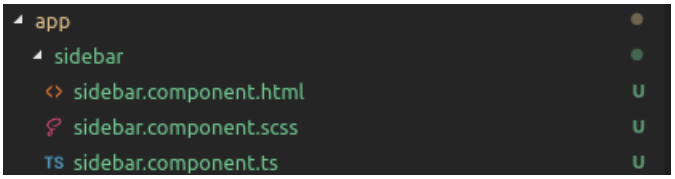


FIGURE 4.3: component

5. Inside the HTML and SCSS file we have written the following code:
   HTML:

```
<nav>
 <ul>
   <li>
     <a routerLink="">
         <i class="material-icons">tab</i>
     </a>
   </li>
   <li>
      <a routerLink="">
        <i class="material-icons">add_circle</i>
      </a>
    </li>
 </ul>
</nav>
```

LISTING 4.1: HTML

SCSS:

```
nav {
   background: #2D2E2E;
   height: 100%;
   position: fixed;
   ul {
       list-style-type: none;
       padding: 0;
       margin: 0;
       li {
           a {
               color: #fff;
               padding: 20px;
               display: block;

           }
           .activated {
               background-color: #00a8ff;
           }
```

Listing 4.2: Scss

6. Now, we need to use Icon Material library which is created by Angular to provide to the developers a wide variety of icons that they can use in their project.In our case, we had the need for such a library so we decided to add it to our project. Now, to add the icon material library we first download the Icon Material library using this command:

```
npm install --save @angular/material
```

Then we will add the Icon Material library to the app.module in order for it to be loaded into our project and be able to use it.

```
import {MatIconModule} from '@angular/material/icon';
 imports: [
   MatIconModule,
```

Finally, we need to add the link for the CSS file of the Icon Material library to the index file so that it can be accessed by every single component in our project.

```
 <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
```

now if we load the page we will get this:

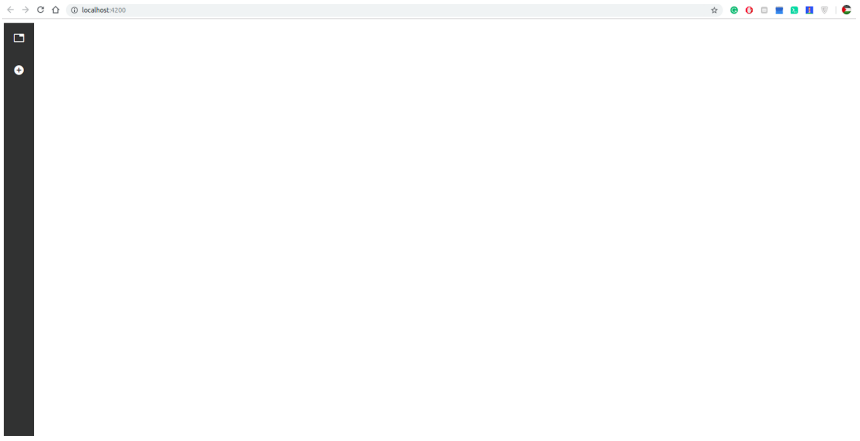


FIGURE 4.4: Sidebar

7. Finishing the design part is great, but right now nothing will happen when we click any of these buttons.So we need to create the routing in order for it to take us to the right page when we click.we will start with creating both the Add and the Display component:

```
ng generate component add
ng generate component display
```

8. next we will create app-routing.module.ts. App-routing.module.ts is an Angular file that takes care of the routing for us but unlike other Angular files it's not automatically created when you create the Angular project, so we need to create it ourselves.Inside the file we will specify the routing using the following code:

```
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import {AddComponent} from './add/add.component';
import {DisplayComponent} from './display/display.component';
const routes: Routes = [
 {
   path: '',
   component:DisplayComponent
 },
 {
   path:'form/add',
   component:AddComponent
 },
 {
   path:'form/edit',
   component:AddComponent
 },
];
@NgModule({
 imports: [RouterModule.forRoot(routes)],
```

```
 exports: [RouterModule]
})
```

Listing 4.3: TS

And in the sidebar.html we will use routerLink to tell angular to which page to route us when the button is clicked

```
      <a routerLink="">
      <a routerLink="form/add">
```

And when we recompile the code we will get the following:

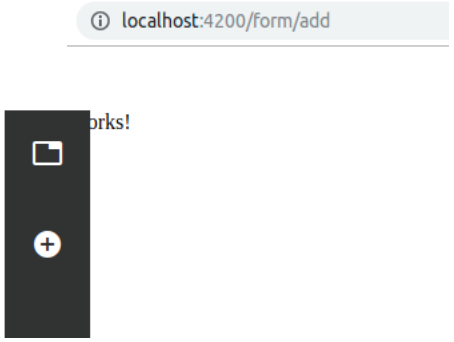


Figure 4.5: Error

9. Its clear from the picture that we have a small problem.the Sidebar will cover the content of the page due to being the parent component that is called by the app*Add*? component, so in order to solve this problem we can just change the SCSS of the Add component to make it appear, but this approach is not good because we will have to do it for every component that we create, so we decided to change the code inside the style.scss since the style code in the master stylesheet will apply to all of the other stylesheets.
   CSS:

```
@import '~@angular/material/prebuilt-themes/deeppurple-amber.css';
body {
   margin: 0;
   background: #F2F2F2;
   font-family: 'Montserrat', sans-serif;
   height: 100vh;
}
#container {
  display: grid;
  grid-template-columns: 70px auto;
  height: 100%;
  #content {
      padding: 30px 50px;
      ul {
          list-style-type: none;
```

```
margin:0;padding:0;
li {
    background: #fff;
    border-radius: 8px;
    padding: 20px;
    margin-bottom: 8px;
    a {
        font-size: 1.5em;
        text-decoration: none;
        font-weight: bold;
        color:#00A8FF;
    }
    ul {
        margin-top: 20px;
        li {
            padding:0;
            a {
                font-size: 1em;
                font-weight: 300;}}}}
```

Now everything will work just fine:

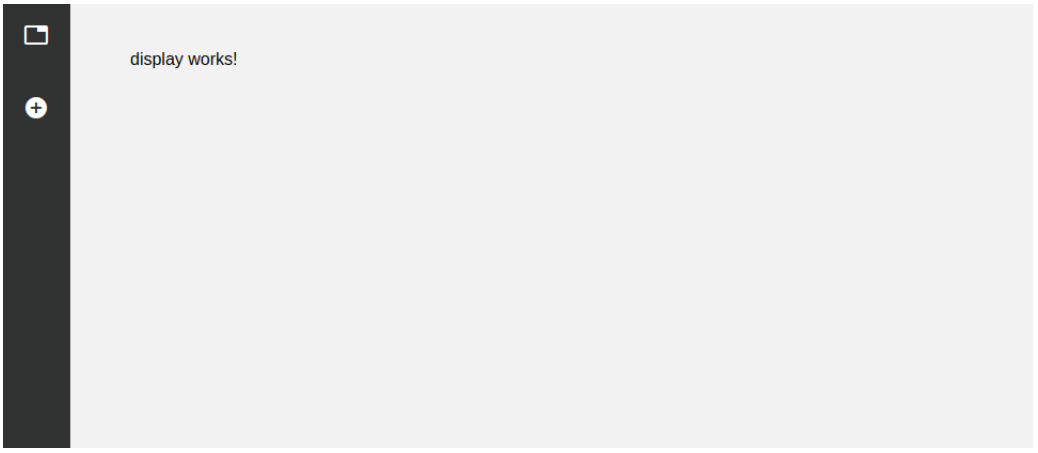


FIGURE 4.6: Error

10. Now we will implement the Display and the Add pages: For the display page we will use card material to show all the rooms and their details but since we will get all the information from the Realtime Databae, we need to connect our application to the Firebase Database. In order to do so we need to first to download Firebase admin using the following command:

```
npm install --save firebase-admin
```

After that we need to create a new Firebase project to be able to use the Realtime Database services. After we create the Firebase project, we need to change the

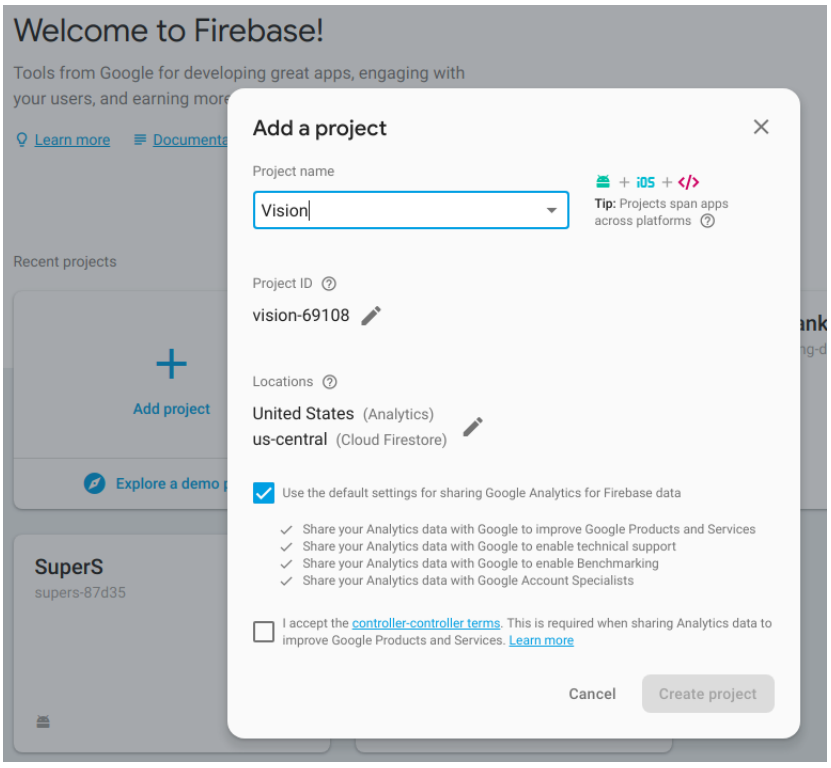


FIGURE 4.7: Firebase

rules so that we can read and write data from and to to our Realtime Database. We navigate to Firebase project rules and we change the promotion to true, which will allow everyone to read and write from out Realtime Database. This is not secure but since the application is under development we have nothing to worry about for now.

```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

FIGURE 4.8: rules

Next, we need to download the private key that Firebase generates for us to be able to use the SDK:

```
var admin = require("firebase-admin");

var serviceAccount = require("path/to/serviceAccountKey.json");

admin.initializeApp({
  credential: admin.credential.cert(serviceAccount),
  databaseURL: "https://vision-6855f.firebaseio.com"
});
```

Unfortunately, this did not work because the Admin SDK cannot be deployed on a client-side web frameworks such as Angular, but rather work only be used in trusted server-side environments. So we had two options in our hands to slove this dilemma, either creating a backend server with a Rest API to host firebase on it, or we can use a third party library that handles the server-side business for us. obviously, we have chosen the second solution since it is more convenient for a small application like ours. To download the library we use the following command:

```
npm i firebase angularfire2
```

Then, we need to add the following code to the environment.ts so we can access all the services provided by Firebase:

```
export const environment = {
 production: true,
 firebase: {
   apiKey: "AIzaSyAQHmkfWYq5xisZ6K2jXvbU08xLyNj0_Do",
   authDomain: "vision-6855f.firebaseapp.com",
   databaseURL: "https://vision-6855f.firebaseio.com",
   projectId: "vision-6855f",
   storageBucket: "vision-6855f.appspot.com",
   messagingSenderId: "521649909109"
 }
};
```

Finally, we import the library inside app.module:

```
import { AngularFireDatabaseModule } from 'angularfire2/database';
import { environment } from '../environments/environment';
 imports: [
   AngularFireModule.initializeApp(environment.firebase),
   ],
```

Now we can read and write to the REaltime Database from our application and we can now start implementing the Add and Display pages.

4.1.1.1 Add/Edit

Inside the Add page we will allow the end user to add his/her name and description of room / rooms. For that we have used a Stepper Form from Angular to ask the user to enter all the details related to the room / rooms. As always, before we can start using the Material we need to add it to our app.modul file with other dependencies such as the animation module to allow animation between each step.

```
import {MatStepperModule} from '@angular/material/stepper';
 imports: [
   MatStepperModule,
   MatFormFieldModule,
   MatInputModule,
   BrowserAnimationsModule,
   ],
```

After we imported the necessary libraries, we can start writing our HTML code. As we can see from the code we wired the input to the HTML field using ngmodel which binds the field value to a variable in our code.

```
<ul>
 <li *ngFor="let room of rooms ">
   <!-- Delete -->
   <mat-card-actions>
     <button  (click)="delete(room.RoomName)" mat-mini-fab color="primary"><i class="materi
     </button>
   </mat-card-actions>
   <!-- Edit -->
   <mat-card-actions>
     <button  mat-mini-fab color="warn" routerLink="form/edit/{{room.RoomName}}" ><i class=
   </mat-card-actions>
   <!-- RoomName -->
   <span id="info">
       <mat-card-header>
       <div mat-card-avatar class="example-header-image">
       </div>
       <mat-card-title class='htmlquestion'[innerHTML]=
       'room.RoomName'>
       </mat-card-title>
     </mat-card-header>
   </span>
   <!-- Descrption -->
   <span class=def>Descrption:</span>
   <span [innerHTML]='room.Descrption' ></span>
   <br>
 </li>
</ul>
```

The page will look like this: After we finish designing the page, we will need to

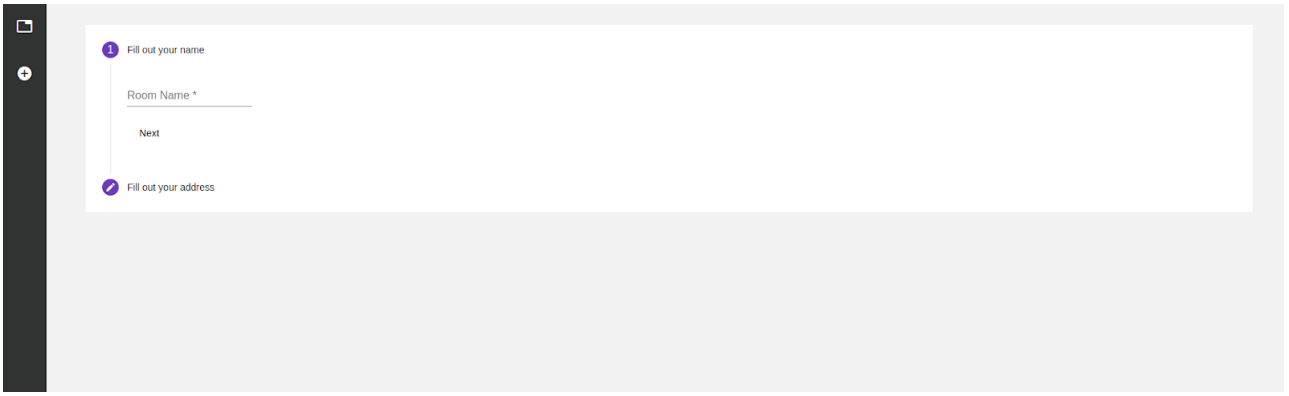


FIGURE 4.9: Add/Edit

add the functionality. First we need to differentiate between the edit page and the create new page. For that we have used ActivatedRoute library which will allow as to access the params of the routes so we can indicate which page are we going to. Inside the ngOnInit which will be called on the create of the page, we will check if the mode == add we create an empty room object and set the mode to new,if not we will retrieve the object by its id from the Firebase.

```
ngOnInit() {
   this.route.params.subscribe((params) => {
     if (params.mode == 'add') {
       this.mode = "New";
       this.room = < Room > {};
       }else if(params.mode == 'edit')  {
       this.mode = "Edit";
       this.room = < Room > {};
       var object = this.db.object(params.id)
       object.snapshotChanges().subscribe(action => {
         this.room.RoomName=action.payload.val()["RoomName"]
         this.room.Descrption=action.payload.val()["Descrption"]
       });
     }
   });
 }
```

Finally, we implemented the functionality of the Done button which will save the object to our Realtime Database and route us back to the display page.

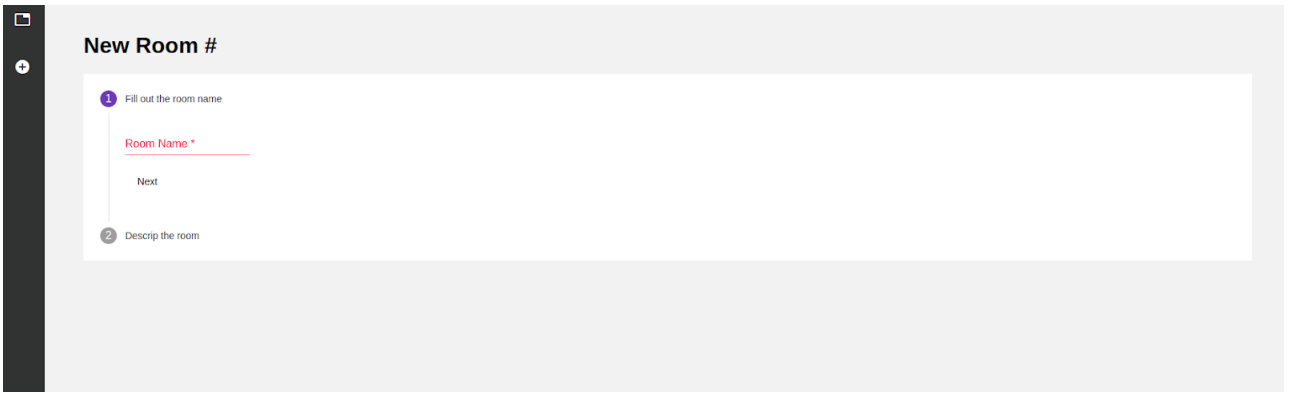


FIGURE 4.10: ADD

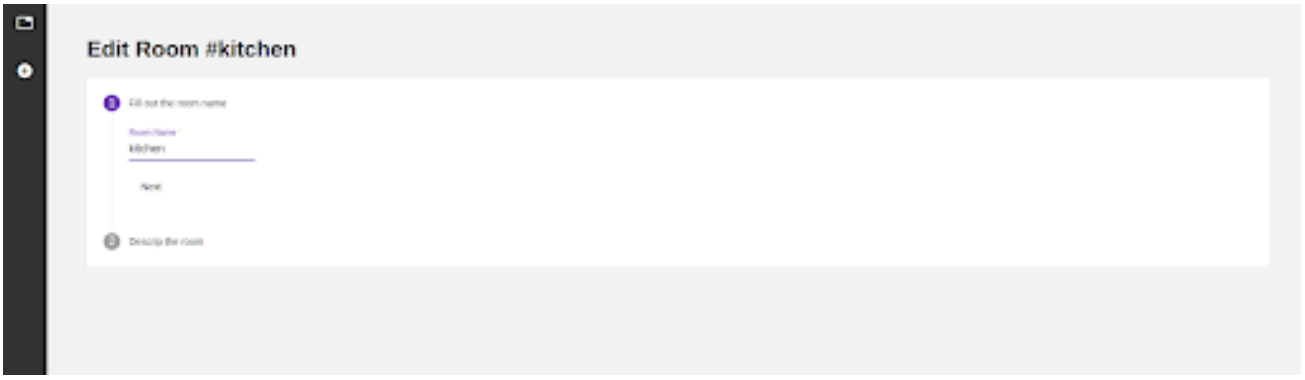


FIGURE 4.11: Edit

4.1.2 Display

To display the information we have used cards component that contain the name of the room and description of each room. We have also used ngfor to iterate over the array of objects and we have created a couple of buttons to allow as to delete or edit the objects.

```
<ul>
 <li *ngFor="let room of rooms">
   <!-- Delete -->
   <mat-card-actions>
     <button  (click)="delete(room._id)" mat-mini-fab color="primary"><i
     class="material-icons">delete</i></button>
   </mat-card-actions>
   <!-- Edit -->
   <mat-card-actions>
     <button  mat-mini-fab color="warn" routerLink="/form/edit/{{room._id}}"
     ><i class="material-icons">edit</i></button>
   </mat-card-actions>https://www.overleaf.com/1548993422cgztmfpkkggk
   <!-- RoomName -->
   <span id="info">
       <mat-card-header>
       <div mat-card-avatar class="example-header-image"></div>
       <mat-card-title class='htmlquestion'[innerHTML]='room.RoomName'></mat-card-title>
     </mat-card-header>
   </span>
       <!-- Descrption -->
       <span class=def>Descrption:</span>
       <span [innerHTML]='room.Descrption' ></span>
       <br>
 </li>
 </ul>
```

To retrieve the objects for the Realtime Database, we have created an array of type room and a room object .

```
    rooms: Room[]=[];
  room: Room;
```

And inside the ngOnInit we get all the objects in a json format, then we iterate over each key using for in loop, after that we store the values in the array.

```
  public delete(id){
  if (confirm("Are you sure you want to delete Room:" + id + '?')) {
    var name = this.db.list('/');
    name.remove(id);
    this.rooms.splice(this.rooms.indexOf(id), 1)
  }
ngOnInit() {
  var object = this.db.object('/')
  object.snapshotChanges().subscribe(action => {
    var array=action.payload.val();
    for (var property in array) {
      this.room = < Room > {};
      this.room.RoomName=action.payload.val()[property]["RoomName"]
      this.room.Descrption=action.payload.val()[property]["Descrption"]
      this.rooms.push(this.room)}
```

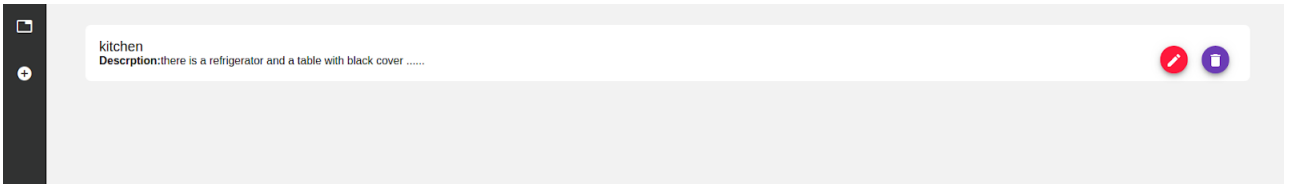


FIGURE 4.12: Edit

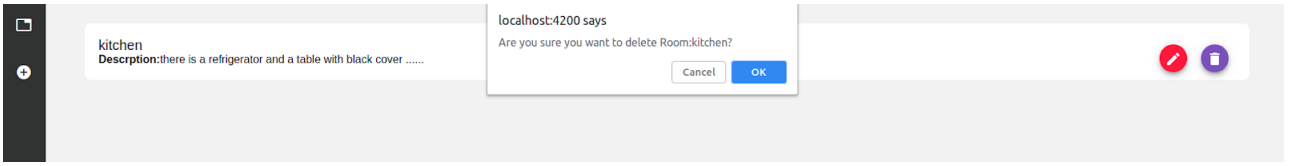
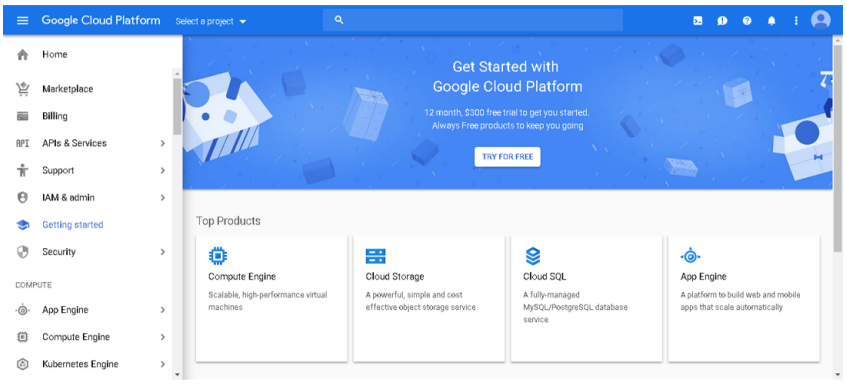


FIGURE 4.13: Edit

4.1.3 Google Vision API

Since we have the Alexa skill working and the Angular app running, the last step is to add the dynamic description of the room / romms. For that we need to use Google cloud Vision API to provide us with dynamic description of the room / rooms.

1. In order to use the vision API that Google provides, we need first to have a developer's account in Google Cloud only then we will be able to create a new project.We start by logging in into our Google cloud Account.



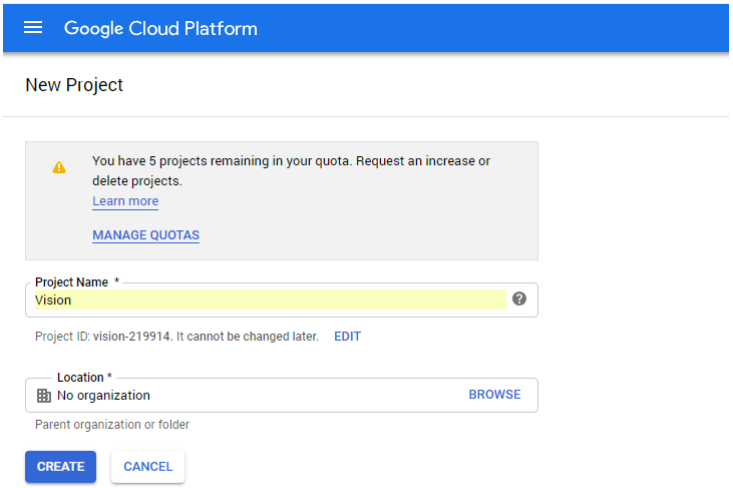


FIGURE 4.14: New project

2. When we hit create,our project will start deploying, and when it's done we will be able to see all the project information and activity.

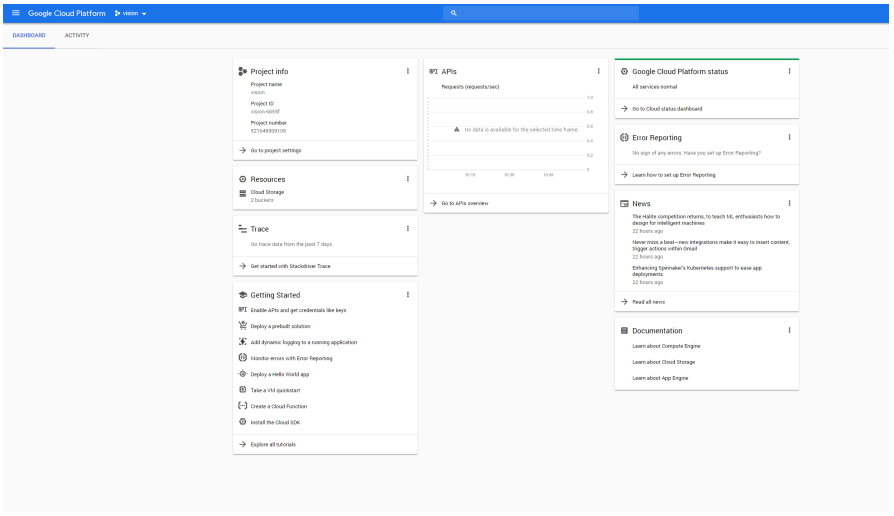


FIGURE 4.15: Project page

3. Next we need to enable the Vision API for the newly created project.

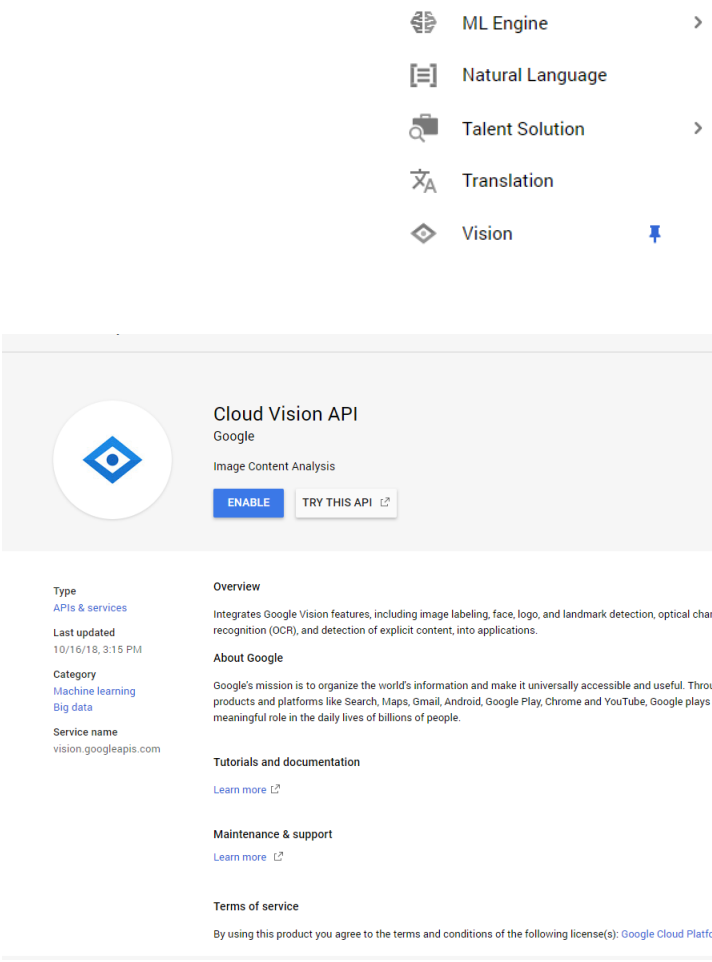


FIGURE 4.16: Enabling Google Vision API

4. Here, we can see all the traffic of our Cloud Vision API.

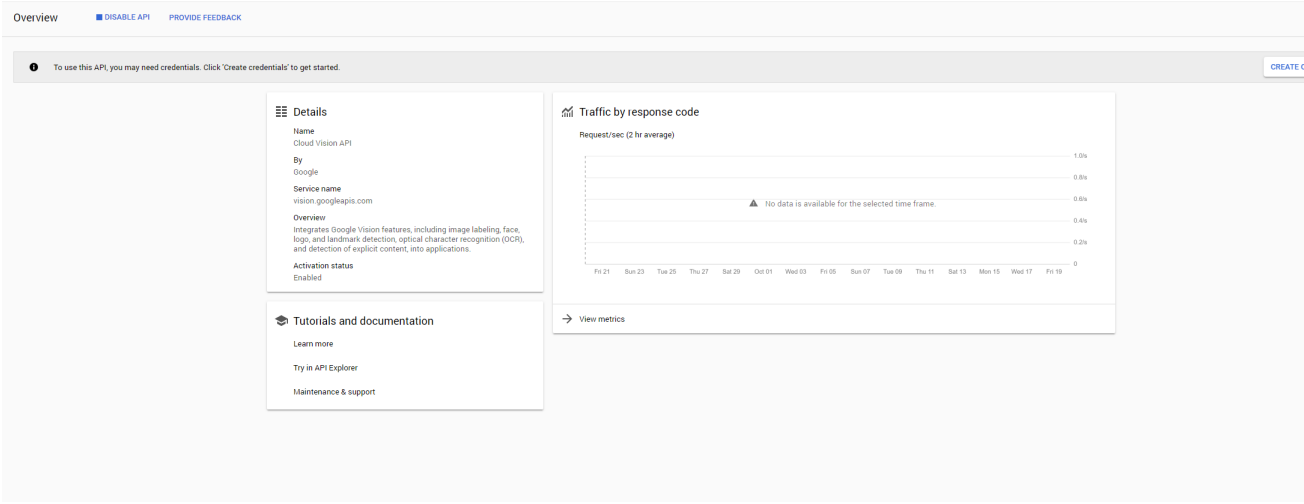


FIGURE 4.17: Google Vision API Traffic

5. After we set up the Vision API, we need to set the credentials in order for the API to function properly. In our case, we require a service account key so we select it and we set the key type to a JSON file.

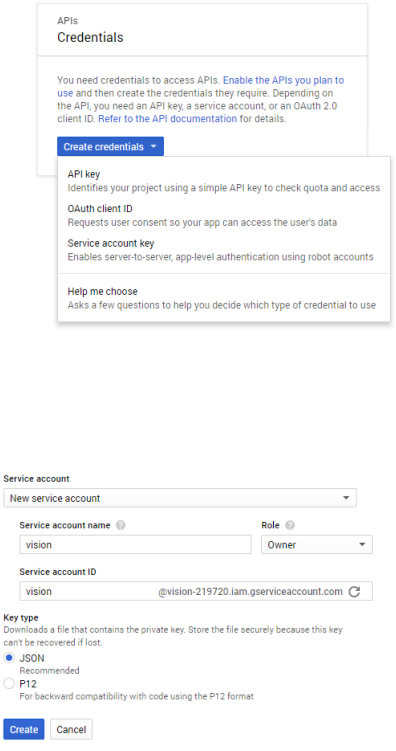


FIGURE 4.18: Setting Credentials

6. Now, we have the key which is a JSON file saved in our system. In order to use the Vision API in our Respeaker board.

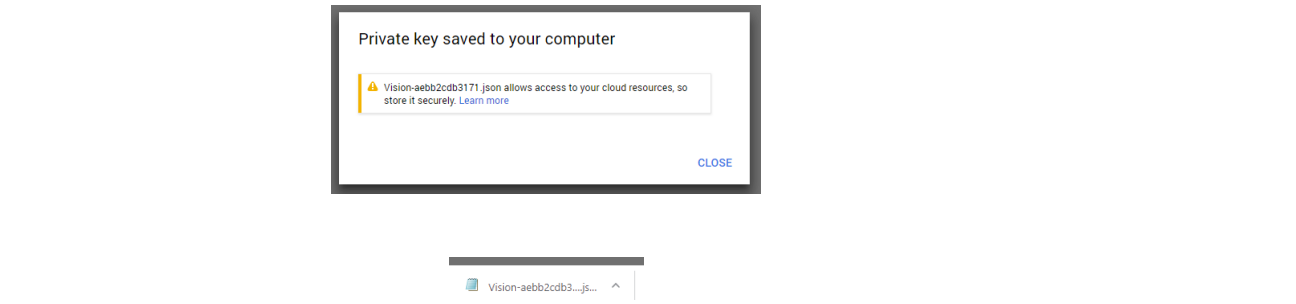


FIGURE 4.19: Setting The Key

After we have finished setting up the Google Cloud, we need to set up the web cam and connect it into our Respeaker board. In our case, we used a GoPro cam.

FIGURE 4.20: GoPro Cam

In order to set up the GoPro on our Respeaker board, we first need to install the fswebcam library.

```
sudo apt-get install fswebcam
```

After we install the library, we use the following command to take a picture.

```
fswebcam image.jpg
```

Now we are ready to do object detection using Google Cloud API, but first we need to have a link between the API and the image that we wish to be processed, of course the best fit is to use Firebase. So, we need to setup Firebase for Python in order to be able to read and write form the cloud Database.

1. We begin by creating our Firebase Project, after that we navigate to the Service Accounts in the project's setting page, then we generate a private key using the button Generate New Private Key which will result in us downloading the key file in JSON format that contains the service account's credentials.

2. Now, we install the SDK, in our case its going to be Python SDK. The following code is used to install the SDK.

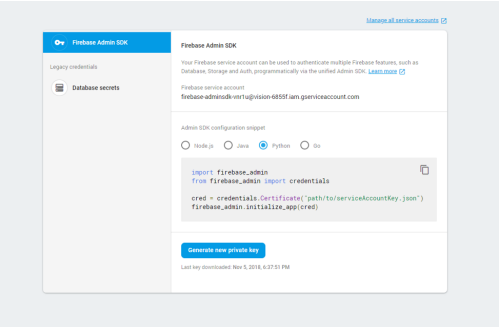


FIGURE 4.21: Account's Credentials

```
sudo pip install firebase-admin
```

3. Next, we Initialize the SDK using the following code.

```
import firebase_admin
from firebase_admin import credentials

cred = credentials.Certificate('path/to/serviceAccountKey.json')
default_app = firebase_admin.initialize_app(cred)
```

4. Last but not least, we installed Outh2 library using the following command.

```
sudo pip install --upgrade oauth2client
```

After setting up the connection to the Firebase, we used Google Cloud Vision API for Labeling taken images and preform a face detection. The code for the labeling is written downward.

```
"""
Google Vision API Tutorial with a Raspberry Pi
and Raspberry Pi Camera.  See more about it here:  https://www.dexterindustries.com/howto/use-
google-cloud-vision-on-the-raspberry-pi/

Use Google Cloud Vision on the Raspberry Pi to take
a picture with the Raspberry Pi Camera and classify it
with the Google Cloud Vision API.   First,
we'll walk you through setting up the Google Cloud Platform.
Next, we will use the Raspberry Pi Camera to take
a picture of an object, and then use the Raspberry Pi
to upload the picture taken to Google Cloud.
We can analyze the picture and return labels
(what's going on in the picture), logos
(company logos that are in the picture) and faces.
```

```
This script uses the Vision API's label detection
capabilities to find a label
based on an image's content.

"""

import argparse
import base64
import os
import json

from googleapiclient import discovery
from oauth2client.client import GoogleCredentials

def takephoto():
   os.system('sudo fswebcam  --no-banner image.jpg')

def main():
    takephoto() # First take a picture
    """Run a label request on a single image"""

    credentials = GoogleCredentials.get_application_default()
    service = discovery.build('vision', 'v1',
    credentials=credentials)

    with open('image.jpg', 'rb') as image:
        image_content = base64.b64encode(image.read())
        service_request = service.images().annotate(body={
            'requests': [{
                'image': {
                    'content': image_content.decode('UTF-8')
                },
                'features': [{
                    'type': 'LABEL_DETECTION',
                    'maxResults': 10
                }]
            }]
        })
        response = service_request.execute()
        print (json.dumps(response, indent=4, sort_keys=True))
        #Print it out and make it somewhat pretty.

if __name__ == '__main__':

    main()
```

This other code is for face detection.

```
"""
Google Vision API Tutorial with a Raspberry Pi and
Raspberry Pi Camera.  See more about it here:  https://www.dexterindustries.com/howto/use-
google-cloud-vision-on-the-raspberry-pi/

Use Google Cloud Vision on the Raspberry Pi to take
a picture with the Raspberry Pi Camera and classify it
with the Google Cloud Vision API.   First, we'll walk
you through setting up the Google Cloud Platform.
Next, we will use the Raspberry Pi Camera to take a
picture of an object, and then use the Raspberry Pi
to upload the picture taken to Google Cloud.
We can analyze the picture and return labels
(what's going on in the picture), logos
(company logos that are in the picture) and faces.

This script uses the Vision API's label
detection capabilities to find a label
based on an image's content.

"""
import os
import argparse
import base64
import json

from googleapiclient import discovery
from oauth2client.client import GoogleCredentials

def takephoto():
   os.system('sudo fswebcam  --no-banner image.jpg')


def main():
    takephoto() # First take a picture
    """Run a label request on a single image"""

    credentials = GoogleCredentials.get_application_default()
    service = discovery.build('vision', 'v1',
    credentials=credentials)

    with open('image.jpg', 'rb') as image:
        image_content = base64.b64encode(image.read())
        service_request = service.images().annotate(body={
            'requests': [{
                'image': {
                    'content': image_content.decode('UTF-8')
                },
                'features': [{
                    'type': 'FACE_DETECTION',
                    'maxResults': 10
                }]
            }]
        })
```

```
        response = service_request.execute()
        print json.dumps(response, indent=4, sort_keys=True)
        #Print it out and make it somewhat pretty.
if __name__ == '__main__':

    main()
```

4.1.4 Skill

So our goal here is to create a skill in order to allow the user to interact with the voice assistance, also we have to script write all the possible narration flows of the conversation that would take place between the user and the voice assistance.

4.1.5 Alexa Skills Kit

Before coding our skill, we need to use Alexa Skills Kit to specify the invocation name of the skill and the word that the assistant should listen to, which will navigate us to the specific intent in the code later on.

1. The first step is to create a new skill in the developer console

FIGURE 4.22: Create A New Skill

2. Next, we set up our skill's invocation name

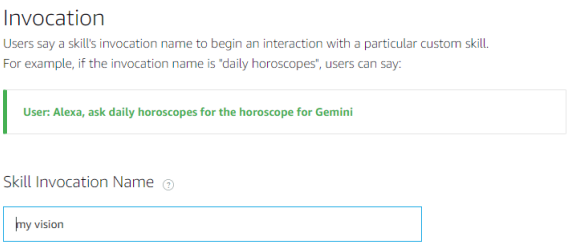


FIGURE 4.23: Skill's Invocation Name

3. After that, we create a new intent for the question of room description where all different grammatical forms of the same question will be stored so that it can pick up the request of description from the user later on.

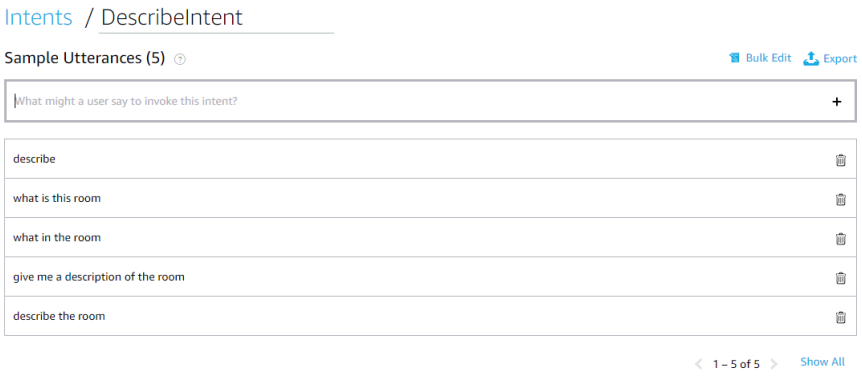


FIGURE 4.24: Describe Intent

4. Then, we create another intent for the question of current number of people in the room.

FIGURE 4.25: Count Intent

5. And last but not least, we specify the endpoint for our skill

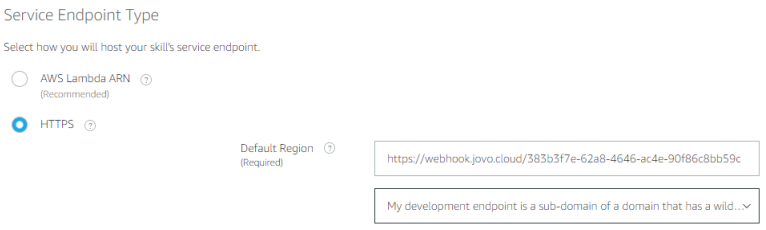


FIGURE 4.26: HTTP Endpoint

4.1.6 Jovo Endpoint

In this part we will implement the Jovo endpoint for our Vision skill, which means that we will write a code that reply to each request that comes from the user.

- Create a new project

```
jovo new Vision
```

  This will give us the default skill code from Amazon

- Generate the Alexa skill JSON file

```
jovo init
```

- Build the project

```
jovo build
```

- Deploy the project

```
jovo deploy
```

  This will connect the Alexa developer kit that we created in the previous section to our jovo skill.

- Install Firebase

```
npm install firbase-admin-sdk
```

4.1.7 Setup

- We start by setting up our Firebase connection

```
'use strict';

// ================================================
// Firebase
// ================================================
var admin = require("firebase-admin");
var serviceAccount = require("/home/omar/Desktop/vision/vision/
vision-6855f-firebase-adminsdk-vnr1u-1485412502.json");
admin.initializeApp({
```

```
  credential: admin.credential.cert(serviceAccount),
  databaseURL: "https://vision-6855f.firebaseio.com"
});
var room="Al yasat hall"
var db = admin.database();
```

- Then, we set up the Jovo app configuration

```
// ====================================================================
// App Configuration
// ====================================================================

const {App} = require('jovo-framework');

const config = {
    logging: true,
};

const app = new App(config);
```

- first we crarte Launch function

```
app.setHandler({
    'LAUNCH': function() {
        this.ask("Welcome to vision app,let me be your vision",
        "how can i help you?")
        },
```

- After we created the DescribeIntent that we created in alexa skill kit.

```
    'DescribeIntent':async function() {
        var ref = db.ref('/'+room+"/DDescrption");
        var Descrption=await ref.once("value").then((snapshot) =>{
            return snapshot.val();});
        this.tell("Right now you are in the "+room+" inside the khalifha
        award building, "+Descrption,"can i help you with something else?")
    },
```

- Finally we created the CountIntent.

```
    'CountIntent':async function() {
        var ref = db.ref('/'+room+"/count");
        var count=await ref.once("value").then((snapshot) =>{
            return snapshot.val();});
        this.tell("Right now you are in the "+room+"I can see , "+count+" pepole in the roo
    },
});
module.exports.app = app;
```

4.2 Smart Banking

The smart banking system as mentioned before, is an IoT implemented system that is used to help the all people especially the visually impaired to access their banking services in an easier and faster and more secure way, this is done by using the voice interface system (Alexa) by giving it commands to control the banking system. The system can do anything from checking the balance, to transferring money, to pay the bills to many more.
To do that first the user will have to log-in by using their voice, and then after that they can access their banking services, now to make the system even more secure we added a hardware button that is kept by the user to act as a two-factor authentication, this button will be used when the user wants to do an action and this button can be modified to be able to do a combination of presses that suits the user.

4.2.1 Skills

To allow the user to interact with the voice assistance, We need to create a skill. In this skill that we have created, we also write the possible conversation that the user will have with Alexa to access the Smart Banking system, we did that by:

4.2.1.1 Alexa Skills Kit

Before we start coding the skill we need to use Alexa skill kit to specify the invocation name of the skill and the word that the assistant should listen to, to route us to the specific intent in the code later on.

1. Create a new skill inside of the developer console

FIGURE 4.27: Crate a new skill

2. Change the Invocation name

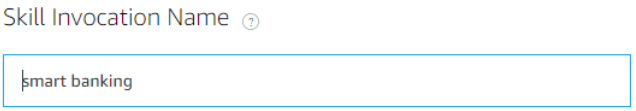


FIGURE 4.28: Invocation

3. Create a new intent to get the balance

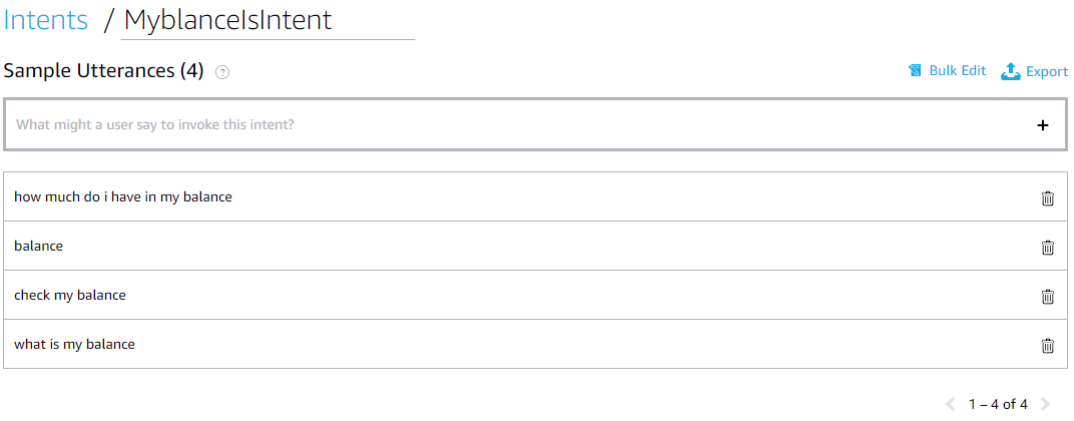


FIGURE 4.29: MybalanceIsIntent

4. Create a new intent to send the money

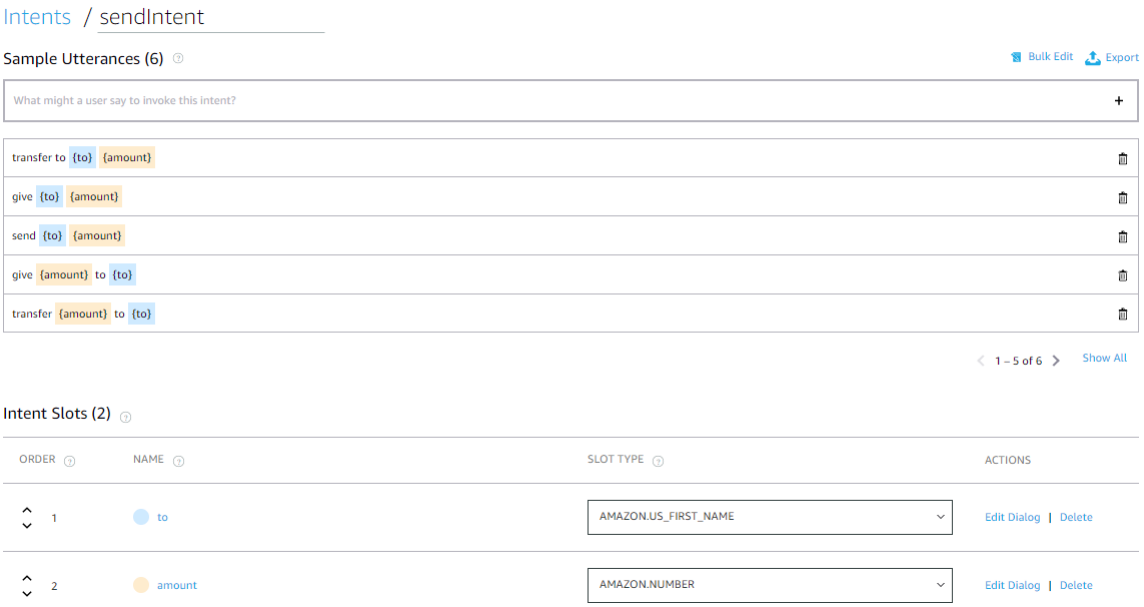


FIGURE 4.30: sendIntent

5. Create a new intent to list transactions

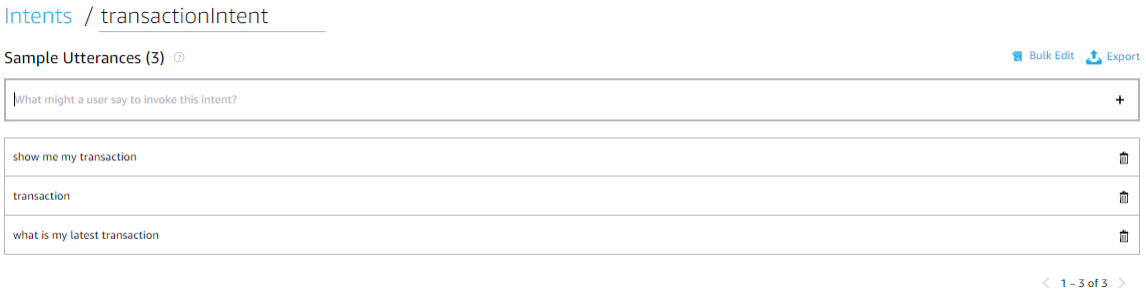


FIGURE 4.31: transactionIntent

6. Create a new intent to order and buy

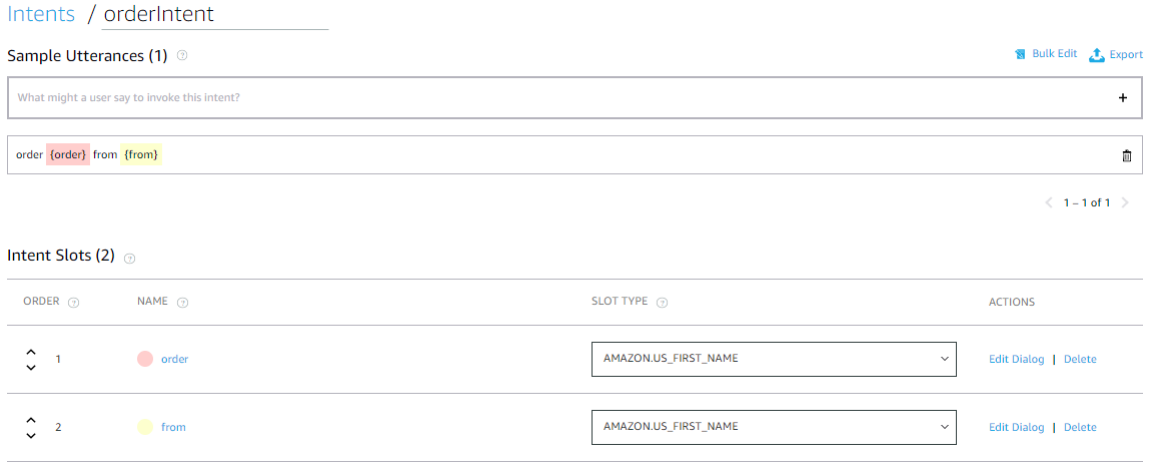


FIGURE 4.32: orderIntent

7. Create a new intent to list orders

FIGURE 4.33: ordersIntent

8. Create a new intent to indicate that the user entered his password

FIGURE 4.34: DoneIntent

9. Create a new intent to Log out

FIGURE 4.35: LogoutIntent

10. Specify the endpoint for the skill

FIGURE 4.36: Endpoint HTTP

4.2.1.2 Jovo Endpoint

In this part we will implement the Jovo endpoint for our skill that we specify in the previous section,which means that we will write a code that reply to each request that the user make, we do that by:

- Create a new project

```
jovo new smartbanking
```

  this will generate for us the default code for Amazon default skill.

- Generate the Alexa skill JSON file

```
jovo init
```

- Build the project

```
jovo build
```

- Deploy the project

```
jovo deploy
```

  This will connect the Alexa developer kit that we created in the previous section to our Jovo skill.

- Install Firebase

```
npm install firbase-admin-sdk
```

After we did all the previous steps we are ready to start coding.

4.2.1.3 Setup and Log-in

- We need to setup our Firebase connection.

```
 'use strict';
// ================================================================================
// Firebase
// ================================================================================
```

```
var admin = require("firebase-admin");
var serviceAccount = require("/home/omar/Desktop/bank/banking/
firebase/smart-banking-d46e6-firebase-adminsdk-uvsrr-a0ff9aef00.json");
admin.initializeApp({
    credential: admin.credential.cert(serviceAccount),
    databaseURL: "https://smart-banking-d46e6.firebaseio.com"
  });
var db = admin.database();
var ref = db.ref();
```

- Setup our Jovo App configuration.

```
// ================================================================================
// App Configuration
// ================================================================================
const {App} = require('jovo-framework');

const config = {
    logging: false,
};

const app = new App(config);
```

- LAUNCH Function.
  this function will start once the user ask Alexa to open the skill. In this function we stored the user id (which is unique for each amazon account) then we retrieve all the data from the database and we check if the userid exists or not, if not we will ask them to contact the bank to register his account so he can start using the skill, then we ask them if he would like to know more about the skill if they answer with no we greet them, and if they say yes, then we provide them with all the information.

```
// ================================================================================
// App Logic
// ================================================================================

 app.setHandler({
     'LAUNCH':async function() {
        var UserId=this.getUserId();
        var Users;
        Users=await ref.once("value").then((snapshot) =>{
            return snapshot.val();});
        var userid=[];
        for (var key in Users) {
                        userid.push(Users[key]["userid"])
        }
        var index=userid.indexOf(UserId)
        if(index > -1)
            this.toIntent("LoginIntent")
        else
```

```
            this.followUpState('NewUser').ask("you
            are not registered with the banking skill,
            to register please contact: 0059915,
            whould you like to know more about banking skill?"
            ,"whould you like to know more about banking skill?")
    },

            'NewUser': {
                    'AMAZON.YesIntent': function() {
                        this.tell("with banking skill you
                        can get full controll over your bank activity
                        using only your voice, for example you can ask me about
                        ,you balance, latest transaction , or even send money")
                    },

                    'AMAZON.NoIntent': function() {
                        this.tell("ok, see you soon")
                    },

    },
```

- Log-in Function

  In the case where we find the user id in our database we will retrieve their name and check if they were logged in or not.if they were logged in, they would be able to use the skill services directly, otherwise Alexa will ask the user to enter their password, after that if the user's password is true then Alexa will reply by asking what kind of service they want to access, on the other hand if the password is incorrect Alexa will then ask the user to renter their password or try again later. We did that by saving the password of the user under their userID.

```
    'LoginIntent':async function() {
       var UserId=this.getUserId();
       var Users;
       Users=await ref.once("value").then((snapshot) =>{
           return snapshot.val();});
       var keys=[];
       var userid=[];
       var Name=[];
       var logged_in=[];
       for (var key in Users) {
                       keys.push(key)
                       userid.push(Users[key]["userid"])
                       Name.push(Users[key]["name"])
                       logged_in.push(Users[key]["logged_in"])
       }
       var index=userid.indexOf(UserId)
       if(logged_in[index]==false){
           db.ref(keys[index]).update({auth:1})
           this.followUpState("enterpassword").ask("Welcome back "+Name[index]+"
           ,please enter your password to login
```

```
            , tell me when you are done
            ,I am going to close my eyes now.","are you done yet?")
        }
        else
            this.ask("hello "+Name[index]+", tell me how can i help you?","how may i help y
    },
            'enterpassword': {
                    'DoneIntent':async function() {
                        var UserId=this.getUserId();
                        var Users;
                        Users=await ref.once("value").then((snapshot) =>{
                            return snapshot.val();});
                        var keys=[];
                        var userid=[];
                        var Name=[];
                        var logged_in=[];
                        var auth=[];
                        for (var key in Users) {
                                        keys.push(key)
                                        userid.push(Users[key]["userid"])
                                        Name.push(Users[key]["name"])
                                        logged_in.push(Users[key]["logged_in"])
                                        auth.push(Users[key]["auth"])
                        }
                        var index=userid.indexOf(UserId)
                        if(auth[index]==2){
                            db.ref(keys[index]).update({logged_in:true})
                            this.ask("you have been authorized to use the smart banking
                            , welcome
                            , how can i help you?"
                            ,"I didn't understand please repeat that")
                        }
                        else if(auth[index]==3)
                            this.tell("sorry you are not authorized because
                            you have entered
                            a wrong password
                            , please try again later")

                        else if(auth[index]==4)
                            this.tell("sorry you are not authorized because you have not
                            entered the password
                            , please try again later")
                        else
                            this.tell("an error has occurred, please contact us")
                            db.ref(keys[index]).update({auth:0})
                        }
    },
```

4.2.1.4 Static Services

In this part of the code, after the user is logged in, Alexa will ask the user of what service they want to use, these are static services which means that they just allow the user to check their balance, transactions or orders without modifying, if that is their intention then Alexa will check the database of the current balance and then return it to the user. Other services that exists, are checking recent transactions, and checking recent orders.

- Balance Service
  In this service we will retrieve the balance and send it back to the user,but since jovo is not a running server we cant just keep storing all the data about the dialog with the user because with each call from the user the server start and then exit, which means all the variables that are saved will be dismissed, so in each intent we need to check if the user is logged-in before we proceed ,if he was not logged in, then we will send them to the log in intent.

```
    'MyblanceIsIntent':async function() {
       var UserId=this.getUserId();
       var Users;
       Users=await ref.once("value").then((snapshot) =>{
           return snapshot.val();});
       var keys=[];
       var userid=[];
       var Name=[];
       var logged_in=[];
       var balance=[];
       for (var key in Users) {
                       keys.push(key)
                       userid.push(Users[key]["userid"])
                       Name.push(Users[key]["name"])
                       logged_in.push(Users[key]["logged_in"])
                       balance.push(Users[key]["balance"])
       }
       var index=userid.indexOf(UserId)
       if(logged_in[index]==true)
           this.ask("hello "+Name[index]+",
           your balance right now is "+balance[index]+"$")
       else
           this.toIntent("LoginIntent")
   },
```

- Transaction Service
  In this service, we will get the latest transaction for the user and then we will ask him if he would live to see more, if his answer is yes, we will then proceed to list all the users transactions.

```
'transactionIntent':async function() {
    var UserId=this.getUserId();
    var Users;
    Users=await ref.once("value").then((snapshot) =>{
        return snapshot.val();});
    var keys=[];
    var userid=[];
    var logged_in=[];
    var transaction=[];
    for (var key in Users) {
                    keys.push(key)
                    userid.push(Users[key]["userid"])
                    logged_in.push(Users[key]["logged_in"])
                    transaction.push(Users[key]["transaction"])
    }
    var index=userid.indexOf(UserId)
    var latest=transaction[index].length-1;
    if(logged_in[index]==true){
        this.followUpState("tran").ask("your latest transaction was
        "+ transaction[index][latest]+"
        , do you want to see more?")
    }
    else
        this.toIntent("LoginIntent")
},
            'tran': {
                'AMAZON.YesIntent':async function() {
                    var UserId=this.getUserId();
                    var Users;
                    Users=await ref.once("value").then((snapshot) =>{
                        return snapshot.val();});
                    var keys=[];
                    var userid=[];
                    var transaction=[];
                    for (var key in Users) {
                                    keys.push(key)
                                    userid.push(Users[key]["userid"])
                                    transaction.push(Users[key]["transaction"])
                    }
                    var tran=""
                    var index=userid.indexOf(UserId)
                    for(var key in transaction[index])
                        tran=tran+","+transaction[index][key]
                    this.ask("here is a list of all your transaction,"+tran)
                },
                'AMAZON.NoIntent': function() {
                    this.ask("yes sir")
},
```

- Orders Service

  Similar to the transaction service we will also show the user his previous orders.

```
'ordersIntent':async function() {
    var UserId=this.getUserId();
    var Users;
    Users=await ref.once("value").then((snapshot) =>{
        return snapshot.val();});
    var keys=[];
    var userid=[];
    var logged_in=[];
    var order=[];
    for (var key in Users) {
                    keys.push(key)
                    userid.push(Users[key]["userid"])
                    logged_in.push(Users[key]["logged_in"])
                    order.push(Users[key]["order"])
    }
    var index=userid.indexOf(UserId)
    var latest=order[index].length-1;
    if(logged_in[index]==true){
        this.followUpState("tran").ask("your latest order was
        "+ order[index][latest]+"
        , do you want to see more?")
    }
    else
        this.toIntent("LoginIntent")
},
            'tran': {
                'AMAZON.YesIntent':async function() {
                    var UserId=this.getUserId();
                    var Users;
                    Users=await ref.once("value").then((snapshot) =>{
                        return snapshot.val();});
                    var keys=[];
                    var userid=[];
                    var order=[];
                    for (var key in Users) {
                                    keys.push(key)
                                    userid.push(Users[key]["userid"])
                                    order.push(Users[key]["order"])
                    }
                    var ord=""
                    var index=userid.indexOf(UserId)
                    for(var key in order[index])
                        ord=ord+","+order[index][key]
                    this.ask("here is a list of all your orders,"+ord)
                },
                'AMAZON.NoIntent': function() {
                    this.ask("yes sir")
                },
},
```

4.2.1.5 Dynamic Services

In this final part, we enabled the users to access the dynamic services, an example of Dynamic Services that we have would be like sending money or ordering items online or paying the bills. If that is the intention of the user then Alexa will comply by sending money to another user by using userIds, but in doing so we opened a door for sensitive information to be manipulated or modified, so to close this door we introduced a two-factor authentication system by using a button that can be used to enter a number of combinations by long or short presses, so if the user wanted to use the dynamic services, Alexa will then ask the user to enter the combination of presses from the button to authorize the user to continue with their service.

- Send Service

  First we will pass user input(slot)that we specify the type of in the Alexa developer kit. for this service we have three inputs, amount(int), to(string), currency(string). so the Alexa sdk will collect these values from the user speech and pass them to the jovo code so it can work with them. The first thing we do in this skill is that we save the transaction information to a temporary place in the database, then we generate a random pattern as a password and ask the user to input it as part of the 2FA process.after the user enters the pattern we will check if it is correct, in the case that it was correct, we will move the transaction information from the temporary location into the final database and we will also withdraw the money from the users account.

```
'sendIntent':async function(amount,to,currency) {
   var UserId=this.getUserId();
   var Users;
   Users=await ref.once("value").then((snapshot) =>{
       return snapshot.val();});
   var size=[3,4,5,6]
   var keys=[];
   var userid=[];
   var logged_in=[];
   var the_pattern="";
   for (var key in Users) {
                  keys.push(key)
                  userid.push(Users[key]["userid"])
                  logged_in.push(Users[key]["logged_in"])
   }
   var index=userid.indexOf(UserId)
   if(logged_in[index]==true){
       for(var i=0;i<size[Math.floor(Math.random()*size.length)];i++){
           the_pattern=the_pattern+Math.round(Math.random())
       }
       var text="send "+amount.value+" "+currency.value+" to "+to.value;
```

```
            db.ref(keys[index]+"/send").update({parttern:the_pattern
            ,sendText:text,amount:amount.value})
            this.followUpState("enterpattern").ask
            ("to "+text+" please enter the following pattern
            ,"+the_pattern
            ,"please say done when you finish")
        }
        else
            this.toIntent("LoginIntent")
    },
            'enterpattern': {
                'DoneIntent':async function() {
                    var UserId=this.getUserId();
                    var Users;
                    Users=await ref.once("value").then((snapshot) =>{
                        return snapshot.val();});
                    var keys=[];
                    var userid=[];
                    var check=[];
                    var sendText=[];
                    var transaction=[];
                    var amount=[];
                    var balance=[]
                    for (var key in Users) {
                                    keys.push(key)
                                    userid.push(Users[key]["userid"])
                                    balance.push(Users[key]["balance"])
                                    check.push(Users[key]["send"]["check"])
                                    sendText.push(Users[key]["send"]["sendText"])
                                    amount.push(Users[key]["send"]["amount"])
                                    transaction.push(Users[key]["transaction"])
                    }
                    var index=userid.indexOf(UserId)
                    var latest=transaction[index].length;
                    if(check[index]){
                        db.ref(keys[index]).update
                        ({balance:balance[index]-amount[index]})
                        db.ref(keys[index]+"/transaction").update
                        ({[latest]:sendText[index]})
                        db.ref(keys[index]+"/send").update
                        ({parttern:"",sendText:""
                        ,check:false,amount:0})
                        this.ask("The transaction"+sendText+" is Completed"
                        ,"How can I help you?")
                    }
                    else
                        this.ask("The transaction"+sendText+" is Denied
                        ,please try again"
                        ,"How can I help you?")
                    }
    },
```

- Order Service

  we did the exact same thing as the send service, the only difference is that this service has to connect to a real provider API in the real implementation.

```
'orderIntent':async function(order,from) {
    var UserId=this.getUserId();
    var Users;
    Users=await ref.once("value").then((snapshot) =>{
        return snapshot.val();});
    var size=[3,4,5,6]
    var keys=[];
    var userid=[];
    var logged_in=[];
    var the_pattern="";
    for (var key in Users) {
                    keys.push(key)
                    userid.push(Users[key]["userid"])
                    logged_in.push(Users[key]["logged_in"])
    }
    var index=userid.indexOf(UserId)
    if(logged_in[index]==true){
        for(var i=0;i<size[Math.floor(Math.random()*size.length)];i++){
            the_pattern=the_pattern+Math.round(Math.random())
        }
        var text="order "+order.value+" from"+to.from;
        db.ref(keys[index]+"/send").update({parttern:the_pattern
        ,sendText:text,amount:"100"})
        this.followUpState("enterpattern").ask
        ("to "+text+" please enter the following pattern
        ,"+the_pattern
        ,"please say done when you finish")
    }
    else
        this.toIntent("LoginIntent")
},
           'enterpattern': {
                'DoneIntent':async function() {
                    var UserId=this.getUserId();
                    var Users;
                    Users=await ref.once("value").then((snapshot) =>{
                        return snapshot.val();});
                    var keys=[];
                    var userid=[];
                    var check=[];
                    var sendText=[];
                    var transaction=[];
                    var amount=[];
                    var balance=[]
                    for (var key in Users) {
                                    keys.push(key)
                                    userid.push(Users[key]["userid"])
                                    balance.push(Users[key]["balance"])
                                    check.push(Users[key]["send"]["check"])
                                    sendText.push(Users[key]["send"]["sendText"])
```

```
                                    amount.push(Users[key]["send"]["amount"])
                                    transaction.push(Users[key]["transaction"])
                        }
                        var index=userid.indexOf(UserId)
                        var latest=transaction[index].length;
                        if(check[index]){
                            db.ref(keys[index]).update(
                            {balance:balance[index]-amount[index]})
                            db.ref(keys[index]+"/transaction").update
                            ({[latest]:sendText[index]})
                            db.ref(keys[index]+"/send").update
                            ({parttern:"",sendText:"",check:false,amount:0})
                            this.ask("The transaction"+sendText+" is Completed"
                            ,"How can I help you?")
                        }
                        else
                            this.ask("The transaction"+sendText+" is Denied,
                            please try again"
                            ,"How can I help you?")
                        }
    },
```

- Log-Out

```
        'AMAZON.StopIntent':async function() {
        var UserId=this.getUserId();
        var Users;
        Users=await ref.once("value").then((snapshot) =>{
            return snapshot.val();});
        var keys=[];
        var userid=[];
        for (var key in Users) {
                        keys.push(key)
        }
        var index=userid.indexOf(UserId)
            db.ref(keys[index]).update({logged_in:false})
            this.tell('I hope to see you again soon.')
    },
    'AMAZON.HelpIntent': function() {
            this.ask('how can i help you?,you can ask me to check you balance
            , or to send money',"how can i help you?")
    },

});

module.exports.app = app;
```

- we run all the code and enter this command

```
 jovo run
```

- Now you can ask Alexa about any services.

4.2.2 2FA Button

In this section we will discuss our implementation of the button.
first we had to setup our project, we did that by:

- add the esp-01 board json file

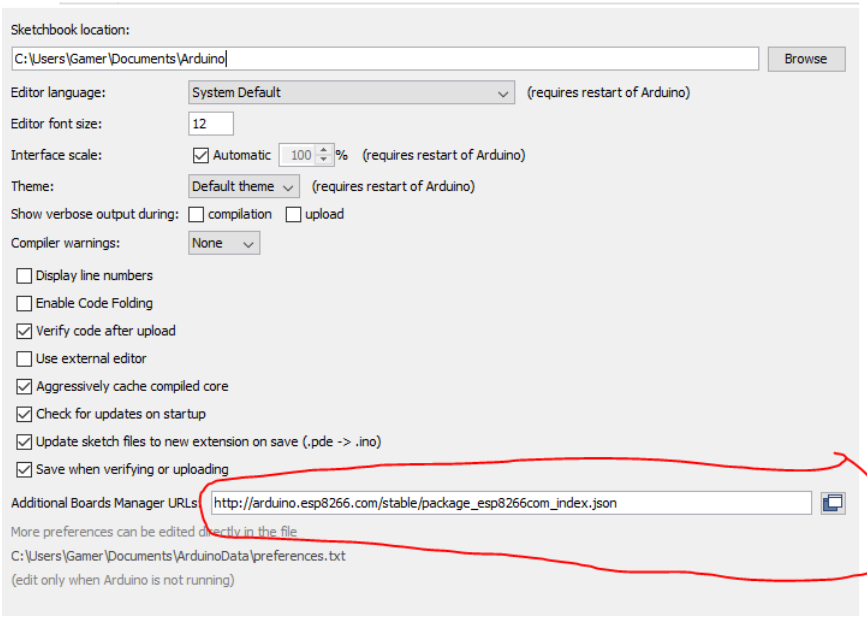


FIGURE 4.37: esp-01 json

- we will install the esp866 boards from board manger

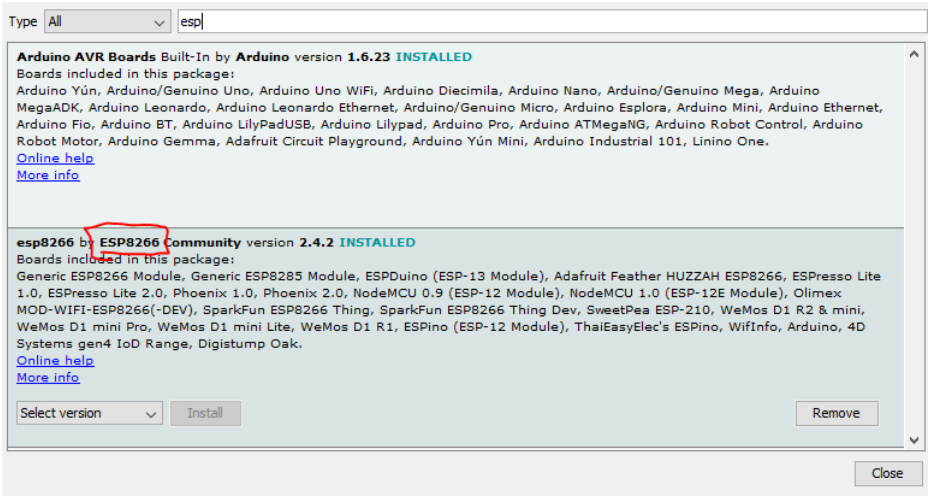


FIGURE 4.38: board manger

- we will select our board.

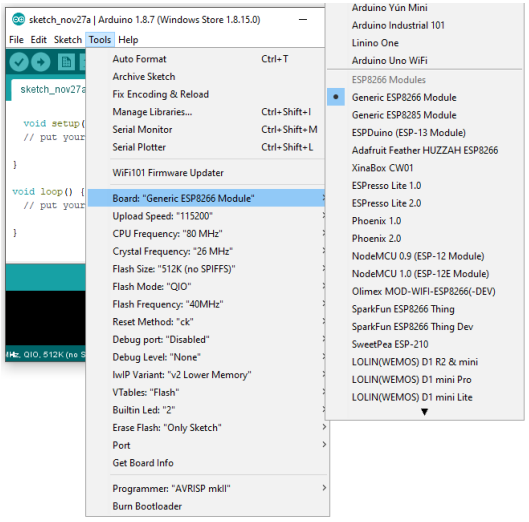


FIGURE 4.39: esp-01

- we will install firebase Arduino library.

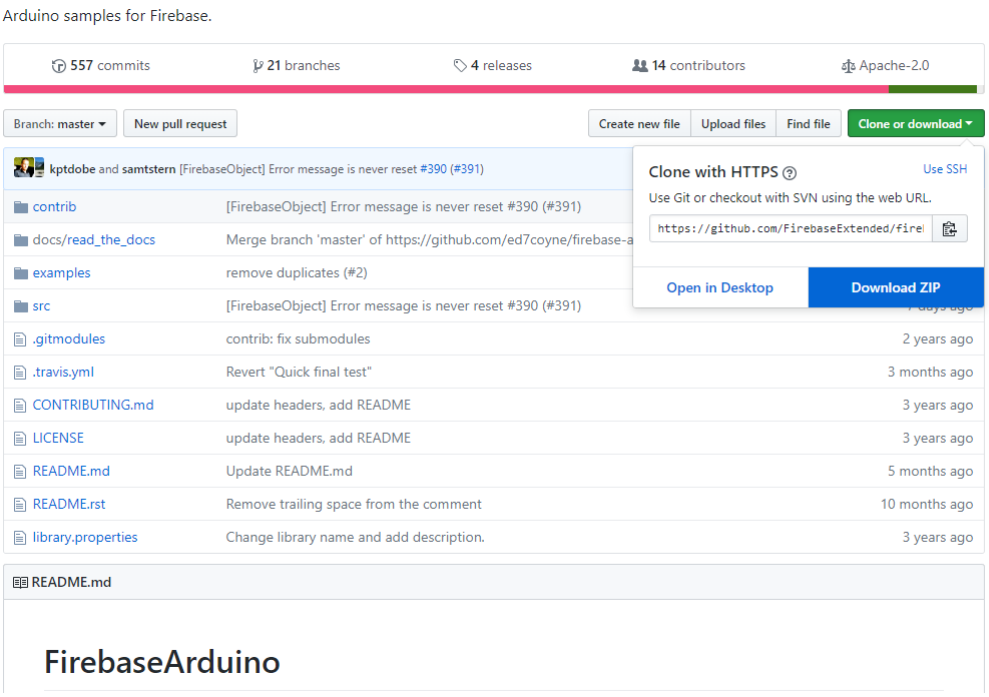


FIGURE 4.40: firebase Arduino

- lastly we will install arduino Json library.

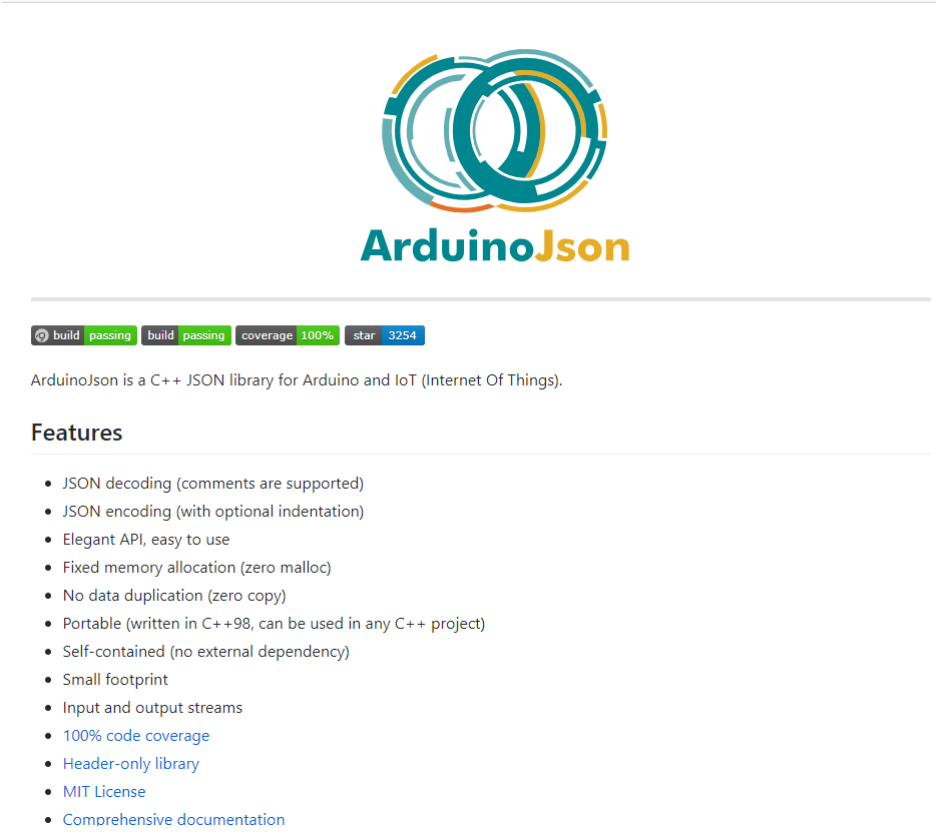


FIGURE 4.41: arduino Json

Now we can start implementing the logic of our smart button.We will start by importing the ESP8266 WiFi and FirebaseArduino library, then we will setup the WiFi and firebase credentials, we will also create a variable that hold the path to our firebase account, and other variables to detect the long and short press.

```
#include <ESP8266WiFi.h>
#include <FirebaseArduino.h>
#define FIREBASE_HOST "smart-banking-d46e6.firebaseio.com"
#define FIREBASE_AUTH "BUwntR3gepiTfiqU43ZKc0Uh3hywUnrbLaX8YMh0"
#define WIFI_SSID "Omar"
#define WIFI_PASSWORD "sinx*cosx"
#define password "omar/password"
#define auth "omar/auth"
#define parttern "omar/send/parttern"
#define check "omar/send/check"
boolean longPressActive = false;
boolean buttonActive = false;
long buttonTimer = 0;
long longPressTime = 250;

String count="";
```

```
int timeout=0;
```

Secondly, we will connect to the WiFi and firebse by using the network name and the password that we defined in the privous step, next we will assigns the pins into input for the button and output for the vibrator motor.

```
void setup() {
  Serial.begin(9600);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  Serial.print("connected: ");

  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  pinMode(2, INPUT);
  pinMode(0, OUTPUT);
}
```

Inside the loop we will keep checking for the auth variable in the firbase and when did it change, because of the loginInted from the jovo section, we will proceed with the vibrator motor and we will keep incriminating the timeout variable, next we will check for the button press, if it was long press we will add to count 1 and if it was short press we will add 0. if the password matches the one that stored in the firbase we will stop the vibrator and send success to the database that allows the user to use the services, if the password was incorrect then we will deny the request.

```
void loop() {
if(Firebase.getInt(auth)==1){
  digitalWrite(0,HIGH);
  timeout++;
  delay(1);
  if (digitalRead(2) == HIGH) {
    if (buttonActive == false) {
      buttonActive = true;
      buttonTimer = millis();
        }
    if ((millis() - buttonTimer > longPressTime)&&(longPressActive == false)) {
        longPressActive = true;
        count=count+"1";
        buttonTimer=0;
        if(Firebase.getString(password)==count){
            Firebase.setInt(auth,0);
            digitalWrite(0,LOW);
            count="";
            timeout=0;
          }
      }
  }
```

```
  else{
    if(buttonActive == true){
      if (longPressActive == true)
        longPressActive = false;
      else{
      count=count+"0";
      longPressActive = false;
      if(Firebase.getString(password)==count){
          Firebase.setInt(auth,0);
          digitalWrite(0,LOW);
          count="";
          timeout=0;
        }
      }
     buttonActive = false;
    }
  }
```

```
  if the user tacks more than 15s to enter the password timeout will occur.
 if(timeout >=200){
  Firebase.setInt(auth,3);
  digitalWrite(0,LOW);
  count="";
  timeout=0;
 }
}
```

We will do exactly the same for any incoming orders.

```
else if(Firebase.getInt(auth)==0 || Firebase.getInt(auth)==3 || Firebase.getInt(auth)==2)
  digitalWrite(0,LOW);
else if(Firebase.getString(parttern)!=""){
  digitalWrite(0,HIGH);
  timeout++;
  delay(1);
  if (digitalRead(2) == HIGH) {
     if (buttonActive == false) {
        buttonActive = true;
        buttonTimer = millis();
        }
  if ((millis() - buttonTimer > longPressTime)&&(longPressActive == false)) {
      longPressActive = true;
      count=count+"1";
      buttonTimer=0;
      if(Firebase.getString(parttern)==count){
          Firebase.setBool(check,true);
          digitalWrite(0,LOW);
          count="";
          timeout=0;
        }
     }
}
```

```
    else{
      if(buttonActive == true){
        if (longPressActive == true)
          longPressActive = false;
        else{
        count=count+"0";
        longPressActive = false;
        if(Firebase.getString(parttern)==count){
            Firebase.setBool(check,true);
            digitalWrite(0,LOW);
            count="";
            timeout=0;
          }
        }
       buttonActive = false;
      }
    }
   if(timeout >=200){
    digitalWrite(0,LOW);
    count="";
    timeout=0;
   }
  }
}
```

4.3 Super Sensor

4.3.1 First Implementation

4.3.2 Sensors

The goal was to monitor all the events that are happening in your house. First,connect all the sensor, collect the data,then we store it in Firebase.

4.3.2.1 BME280

The connect the BME280 its recommended to use SPI for speed.As we can see in Figure 4.42 we wired to sensor to the Feather MO using I2C for simplicity[39].The instruction to connect the sensor to the Feather M0 as follow:

1. Step one, connect from the Feather 3V to the sensor Vin.
2. Step two, connect the Feahter GND to the sensor GND.

3. Step three, connect SCL to the sensor SCK and connect SDA to the sensor SDI.

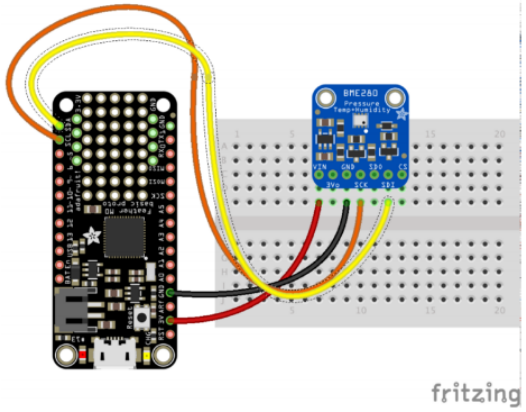


FIGURE 4.42: BME280 Connection

4.3.2.2 LSM9DS1

LSM9DS1 can be used with any micro controller and also it offer I2C and SPI connection[35],we used I2C for minimal connection as we can see in Figure 4.43.The instruction to connect the sensor to the Feather M0 as follow:

1. Step one, connect from the Feather 3V to the sensor Vin.
2. Step two, connect the Feahter GND to the sensor GND.
3. Step three, connect SCL pin to sensor SCL, and connect SDA to the sensor SDA.

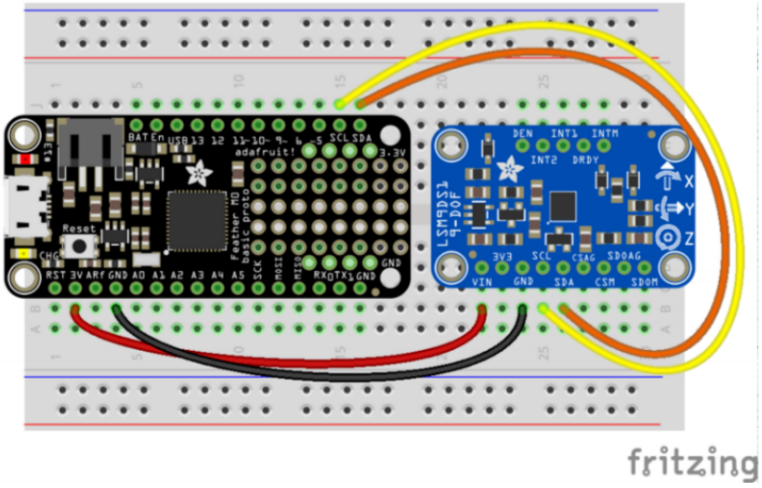


FIGURE 4.43: LSM9DS1 Connection

4.3.2.3 SPH0645

The instruction to connect the sensor to the Feather M0 as follow:

1. Step one, connect board 3V to the sensor 3V.
2. Step two, connect board GND to the sensor GND.
3. Step three, connect TX pin to the sensor BCLK.
4. Step four, connect pin 9 to the sensor DOUT.
5. Step fifth, connect RX pin to the sensor LRCL.

The final connections shown in Figure 4.44

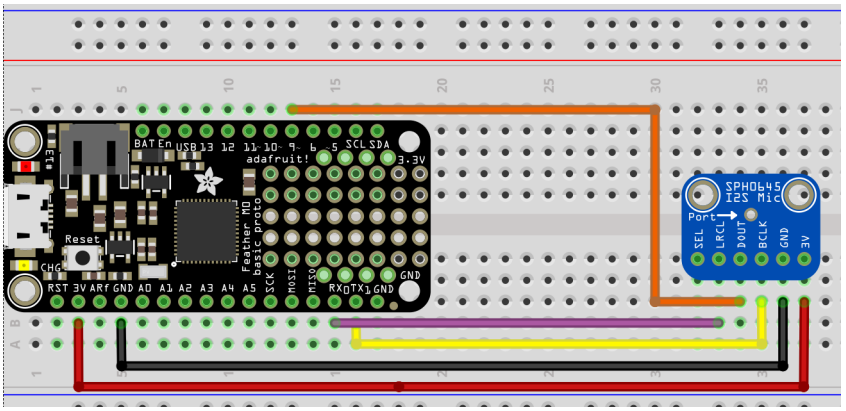


FIGURE 4.44: SPH0645 Connection

4.3.2.4 TSL2561

TSL2561 uses an I2C connection to communicate with the Feather M0 as we can seen in Figure4.45.[40]
The instruction to connect the sensor to the Feather M0 as follow:

1. Step one, connect from the Feather 3V to the sensor Vin.
2. Step two, connect the Feahter GND to the sensor GND.
3. Step three, connect SCL pin to sensor SCL, and connect SDA to the sensor SDA.

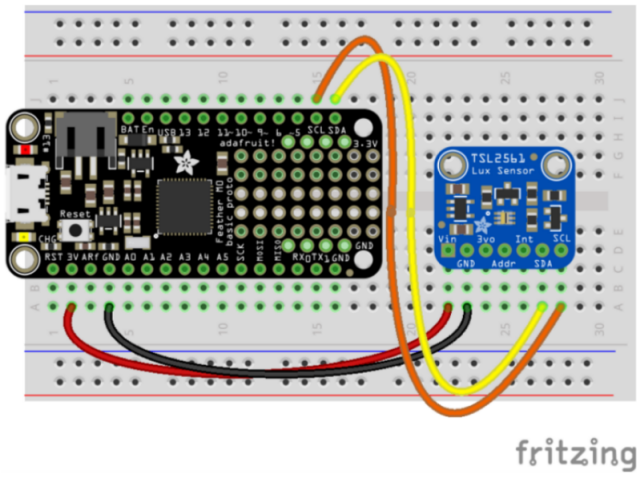


FIGURE 4.45: TSL2561 Connection

4.3.2.5 Collecting the data

In here we simply just getting the reading of each sensor and display it on the serial monitor.The issue was we keep sending the data all time,so this is waste of the space, plus that make the Feather busy.So, we found a smart way to reduce this issue using the code below:

```
//#include <SparkFunLSM9DS1.h>
//#include <I2S.h>
//include libraries for BME280
#include <stdint.h>
#include "SparkFunBME280.h"
//include wire librarie(I2C protocal)
#include <Wire.h>
//include libraries for TSL2561
#include <Adafruit_Sensor.h>
#include <Adafruit_TSL2561_U.h>
//LSM9DS1 imu;

//setup TSL2561
Adafruit_TSL2561_Unified tsl = Adafruit_TSL2561_Unified(TSL2561_ADDR_FLOAT, 12345);
//setup BME280
BME280 mySensor;
void setup() {
  //we set the Serial to 115200 baud
  Serial.begin(115200);
//  I2S.begin(I2S_PHILIPS_MODE, 17000, 32);
//  imu.settings.device.commInterface = IMU_MODE_I2C;
//  imu.settings.device.mAddress = LSM9DS1_M;
//  imu.settings.device.agAddress = LSM9DS1_AG;
//  imu.begin();
```

```
  //we do the settings for BME280(protocol is I2C)
  mySensor.settings.commInterface = I2C_MODE;
  mySensor.settings.I2CAddress = 0x77;
  mySensor.settings.runMode = 3; //Normal mode
  mySensor.settings.tStandby = 0;
  mySensor.settings.filter = 0;
  mySensor.settings.tempOverSample = 1;
  mySensor.settings.pressOverSample = 1;
  mySensor.settings.humidOverSample = 1;
 //we enable communication for the BME280
  mySensor.begin();}
// int i;
 //variable to store the old light sensor value
 int oldlight=0;
 //variable to store the old flame sensor value
 int oldflame=0;
 //variable to store the last time we got a reading from BME sensor
 int bmeTime=0;
 //variable to store the last time we got a reading from light sensor
 int lightTime=0;
 //variable to store the last time we got a reading from flame sensor
 int flameTime=0;
void loop() {
//**************We did not use the mic and the Lms9ds in the final code
because we need a better Machine,that is why we went for Respeaker in the
second Implementation.**************
//int sample = I2S.read();
//imu.readGyro();
//imu.readAccel();
//imu.readMag();
//if ((sample == 0) || (sample == -1) ) {return;}
//sample >>= 14;
//Serial.println("1");
//Serial.println(sample);
//Serial.println("2");
// Serial.print(imu.calcGyro(imu.gx), 2);
// Serial.print(imu.calcGyro(imu.gy), 2);
// Serial.println(imu.calcGyro(imu.gz), 2);
//Serial.println("3");
// Serial.print(imu.calcAccel(imu.ax), 2);
// Serial.print(imu.calcAccel(imu.ay), 2);
// Serial.println(imu.calcAccel(imu.az), 2);
//Serial.println("4");
// Serial.print(imu.calcMag(imu.mx), 2);
// Serial.print(imu.calcMag(imu.my), 2);
// Serial.println(imu.calcMag(imu.mz), 2);
//i++;
//if(i%255==254){
     //create a variable event of type sensors_event_t
     sensors_event_t event;
     tsl.getEvent(&event);
     //get the intensity of the light
     int light=event.light;
     //get the flame sensor value
     int flame=analogRead(A3);
```

```
      //check if the value of the light increased or decreased dramatically and
      5S pass since the last reading
      if((light> oldlight*2 && lightTime==50 ) || (oldlight>light*2 && lightTime==50)){
         //send 1 so the python know that the next value are for the light
         Serial.println("1");
         //send the light value
         Serial.println(light);
         //update the old value
         oldlight=light;
         //reset the time for light
         lightTime=0;
      }
      //check if the value of the flame increased or decreased dramatically and
      5S pass since the last reading
      if((flame>oldflame*2 && flameTime==50) || (oldflame>flame*2 && flameTime==50)){
         //send 2 so the python know that the next value are for the flame
         Serial.println("2");
         //send the flame value
         Serial.println(flame);
         //update the flame value
         oldflame=flame;
         //rest the time for flame
         flameTime=0;
      }
      //if 20S pass(every 20S the update these value)
      if(bmeTime==200){
      //send 3 so the python know that the next value are for the
      TempC,pressure,humidity and altitude
      Serial.println("3");
      //send the the temp
      Serial.println(mySensor.readTempC());
      //send the Pressure
      Serial.println(mySensor.readFloatPressure());
      //send the humidity
      Serial.println(mySensor.readFloatHumidity());
      //send the altitude
      Serial.println(mySensor.readFloatAltitudeMeters());
      //rest the time
      bmeTime=0;
      }
      //increment the time
      bmeTime++;
      lightTime++;
      flameTime++;
      //delay 0.1S(so bmeTime=200 = 20S ,lightTime and flameTime=50 =5S)
      delay(100);
//       }
  }
```

4.3.2.6 Firebase

Afte receiving the data from the Feather,we send it to the firebase.For that we use firebase library in python,but what if the user have more than one sensor in his home and also what if we have Multiusers?.To solve this issue the user should enter the Gmail , username and the room which the sensor in.The code explained below:

```
#Serial(it allow as to collact data  from the serial)
import serial
#firebas(it allow us read and write to firebase)
from firebase.firebase import FirebaseApplication, FirebaseAuthentication
#firebase setting
SECRET = 'pyDYJf7bXrujgAysluibQyuZnRm4qRvV504HbJq6'
DSN = 'https://supers-87d35.firebaseio.com'
EMAIL = 'omar.y.altawil@gmail.com'
authentication = FirebaseAuthentication(SECRET,EMAIL, True, True)
firebase = FirebaseApplication(DSN, authentication)
#initialize a variable arduino that will be listing on COM5
arduino = serial.Serial("COM5", 9600)
#initialize youer gmail that you will acsses from
email="omar7altawil"
#initialize the room that you will setup the super sensor in
location ="livingroom"
# location ="bedroom"
# location ="bathroom"
#location ="kitchen"
"""Receiving data and storing it in a list"""
#run for ever
while True:
    #***************we are supposed to receive my information here and do
    analize for the data and send to the data base the label**************************
                                        #***************but AI part is not
                                    complete yet ************************


 #read the value coming from the arduino
 value=arduino.readline().strip()
 print (value)
 #if we recive 1
 if value=="1":
   #we will store the next value in light variable
   light=arduino.readline().strip()
   #we will post light to firebase to the specific user and location
   snapshot = firebase.patch('/users/'+email+'/'+location, {'light':light})
 #if we recive 2
 elif value=="2":
     # we will store the next value in flame variable
     flame =arduino.readline().strip()
     # we will post flame to firebase to the specific user and location
     snapshot = firebase.patch('/users/' + email + '/' + location, {'flame': flame})
# if we recive 1
 elif value=="3":
     # we will store the next value in TempC variable
     TempC=arduino.readline().strip()
```

```
# we will store the next value in Pressure variable

Pressure=arduino.readline().strip()
# we will store the next value in Humidity variable
Humidity=arduino.readline().strip()
# we will store the next value in Altitude variable
Altitude=arduino.readline().strip()
# we will post all the variable to firebase to the specific user and location
snapshot = firebase.patch('/users/' + email + '/' + location, {'TempC': TempC})
snapshot = firebase.patch('/users/' + email + '/' + location, {'Pressure': Pressure})
snapshot = firebase.patch('/users/' + email + '/' + location, {'Humidity': Humidity})
snapshot = firebase.patch('/users/' + email + '/' + location, {'Altitude': Altitude})
```

4.3.3 Machine Learning

One of the main goals of this project is to be able to sense whats going on in a specific room exactly with just a little chip in hand, so to achieve this we used machine learning technique to train our sensor on the different events that may occur in different situation.We decided to start train our sensor using the mic to be able to sense the water coming out of the faucet and maybe we can train it to calculate how much water has been drooped,and all of that just depending on the sound.The data we have was collected from different people from AI course in the winter.The data was mp3 files as shown in Figure 4.46 for Shower ,Water Flushing,Faucet On,off, Microwave ,Chatterer, etc.

43	Faucet On	60s	307.6		
44	Faucet On-Hand Washing	60s	3.3		
45	Faucet On-Hand Washing	60s	17.6		
46	Faucet On-Hand Washing	60s	42.9		
47	Faucet On-Hand Washing	60s	107.67		
48	Faucet On-Hand Washing	60s	315.93		
49	Shower On	60s	60.76		
58	Water Flushing				
59	Faucet On	60s	5.67		
60	Faucet On	60s	7.96		
61	Faucet On	60s	13.406		
62	Faucet On	60s	27.7		
63	Faucet On	60s	46.15		
64	Faucet On-Dish Washing	60s	5.6		
65	Faucet On-Dish Washing	60s	7.29		
66	Faucet On-Dish Washing	60s	12.7		
67	Faucet On-Dish Washing	60s	28.78		
68	Faucet On-Dish Washing	60s	43.03		
69	Faucet Off (Ambient Sounds - Silence)	60s			
70	Faucet Off (Ambient Sounds - Silence)	60s			
71	Faucet Off (Ambient Sounds - Silence)	60s			
72	Faucet Off (Ambient Sounds - Walking)	60s			
73	Faucet Off (Ambient Sounds - Chatter)	60s			
74	Faucet Off (Ambient Sounds - Blender)	60s			
75	Faucet Off (Ambient Sounds – Washer)	60s			
76	Faucet Off (Ambient Sounds - Silence)	60s			

FIGURE 4.46: Mp3 Files for ML

We train the data provided using PyAudio library.Also we did features extraction for the data so we can know all the events in the room. The code for the ML is :

```
from pyAudioAnalysis import audioBasicIO

from pyAudioAnalysis import audioFeatureExtraction

import matplotlib.pyplot as plt

import numpy as np

import pyaudio

import wave

CHUNK = 10000

FORMAT = pyaudio.paInt16

CHANNELS = 1

RATE = 44100

RECORD_SECONDS = 300

WAVE_OUTPUT_FILENAME = "/Users/mohdghazal/Desktop/output.wav"
```

```
p = pyaudio.PyAudio()

stream = p.open(format=FORMAT,

channels=CHANNELS,

rate=RATE,

input=True,

frames_per_buffer=CHUNK)

print("Starting * recording")

frames = []

plt.axis([0,33,-5,5])

plt.show(block=False)

for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):

data = stream.read(CHUNK)

x = np.fromstring(data, dtype=np.int16)

F = audioFeatureExtraction.stFeatureExtraction(x,RATE, 0.050*RATE, 0.025*RATE)

frames.append(data)

plt.axis([0,33,-5,5])

plt.plot(np.mean(F,axis=1))

plt.draw()

plt.pause(0.001)

plt.clf()

print("* done recording")

stream.stop_stream()

stream.close()

p.terminate()

wf = wave.open(WAVE_OUTPUT_FILENAME, 'wb')

wf.setnchannels(CHANNELS)

wf.setsampwidth(p.get_sample_size(FORMAT))
```

```
wf.setframerate(RATE)

wf.writeframes(b''.join(frames))

wf.close()
```

The results for features extraction is shown in Figure 4.47 and Figure 4.48

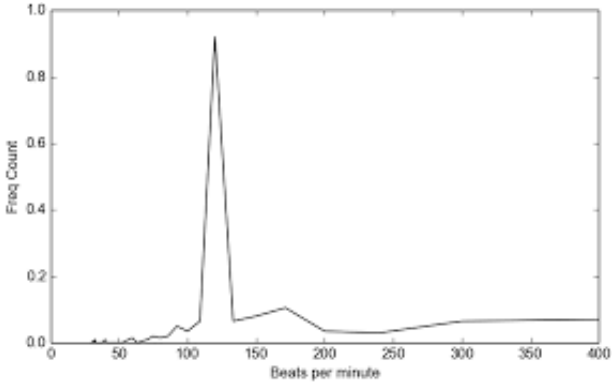


FIGURE 4.47: Features Extraction Results

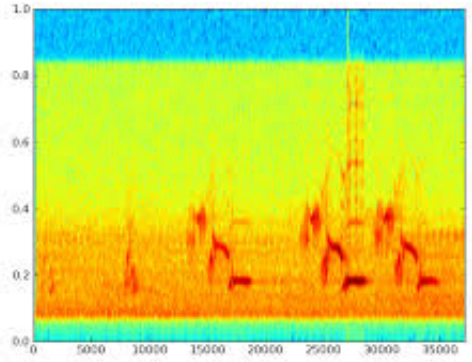


FIGURE 4.48: Features Extraction Results

4.3.3.1 Classification

Now we have the data set ready we need to test different classification to chose the best for our data. We used KNN ,SVM,gradientboosting,extratrees and randomforesst Classifier techniques.The code is as follow:

```
from pyAudioAnalysis import audioTrainTest as aT
aT.featureAndTrain(["/home/tyiannak/Desktop/MusicGenre/Classical/","/home/tyia
nnak/Desktop","/home/tyiannak/Desktop/1"], 1.0, 1.0, aT.shortTermWindow,
aT.shortTermStep, "svm", "svmMusicGenre3", True)
aT.featureAndTrain(["/home/tyiannak/Desktop/MusicGenre/Classical/","/home/tyia
nnak/Desktop","/home/tyiannak/Desktop/2"], 1.0, 1.0, aT.shortTermWindow,
```

```
aT.shortTermStep, "knn", "knnMusicGenre3", True)
aT.featureAndTrain(["/home/tyiannak/Desktop/MusicGenre/Classical/","/home/tyia
nnak/Desktop","/home/tyiannak/Desktop/3"], 1.0, 1.0, aT.shortTermWindow,
aT.shortTermStep, "extratrees", "etMusicGenre3", True)
aT.featureAndTrain(["/home/tyiannak/Desktop/MusicGenre/Classical/","/home/tyia
nnak/Desktop","/home/tyiannak/Desktop/4"], 1.0, 1.0, aT.shortTermWindow,
aT.shortTermStep, "gradientboosting", "gbMusicGenre3", True)
aT.featureAndTrain(["/home/tyiannak/Desktop/MusicGenre/Classical/","/home/tyia
nnak/Desktop","/home/tyiannak/Desktop/5"], 1.0, 1.0, aT.shortTermWindow,
aT.shortTermStep, "randomforest", "rfMusicGenre3", True)
```

Part of the results for the classifications shown in the Figure 4.49

```
Shower(1)
SVM              = [ON =0.05, OFF=0.05 ,Hand Washing=0.02, Dish Washing=0.01 ,Shower=0.87, Flushing=0.0]
KNN              = [ON =1.0, OFF=0.0 ,Hand Washing=0.0, Dish Washing=0.0 ,Shower=0.0, Flushing=0.0]
extratrees       = [ON =0.37, OFF=0.06 ,Hand Washing=0.06, Dish Washing=0.01 ,Shower=0.5, Flushing=0.0]
gradientboosting = [ON =0.23, OFF=0.01 ,Hand Washing=0.0, Dish Washing=0.0 ,Shower=0.76, Flushing=0.0]
randomforest     = [ON =0.37, OFF=0.08 ,Hand Washing=0.06, Dish Washing=0.01 ,Shower=0.47, Flushing=0.0]

Flush(1)
SVM              = [ON =0.19, OFF=0.21 ,Hand Washing=0.07, Dish Washing=0.2 ,Shower=0.16, Flushing=0.17]
KNN              = [ON =0.0, OFF=0.0 ,Hand Washing=0.0, Dish Washing=0.0 ,Shower=0.0, Flushing=1.0]
extratrees       = [ON =0.27, OFF=0.13 ,Hand Washing=0.14, Dish Washing=0.18 ,Shower=0.06, Flushing=0.23]
gradientboosting = [ON =0.2, OFF=0.24 ,Hand Washing=0.3, Dish Washing=0.03 ,Shower=0.21, Flushing=0.01]
randomforest     = [ON =0.26, OFF=0.13 ,Hand Washing=0.17, Dish Washing=0.18 ,Shower=0.09, Flushing=0.17]
```

FIGURE 4.49: The Classifications results

4.3.3.2 Regression

In this part,the goal was to find the total water consumption so using PyAudio we train the data shown in Figure 4.50 using this code:

```
from pyAudioAnalysis import audioTrainTest as aT
aT.featureAndTrainRegression("data/speechEmotion/", 1, 1, aT.shortTermWindow,
aT.shortTermStep, "svm", "data/svmSpeechEmotion", False)
```

The result for regression shown in Figure 4.51.

File Name (Training Data)	Description	Duration	Flow Rate (ml/s)
1	1	60s	33.33333333
2	1	60s	50
3	1	60s	61.81818182
4	1	60s	113.3333333
5	1	60s	136
6	2	60s	35.71428571
7	2	60s	50
8	2	60s	83.33333333
9	2	60s	104.6153846
10	2	60s	141.6666667
11	3	60s	52.30769231
12	3	60s	61.81818182
13	3	60s	85
14	3	60s	104.6153846
15	3	60s	125.9259259
16	4	37s	-
17	4	37s	-
18	4	37s	-
19	4	37s	-
20	4	37s	-
21	1	60s	42.5
22	1	60s	56.66666667
23	1	60s	85
24	1	60s	107.9365079
25	1	60s	170
26	5	60s	25
27	5	60s	55.55555556
28	5	60s	83.33333333
29	5	60s	123.6363636
30	5	60s	194.2857143
31	6	60s	-
32	6	60s	-
33	6	60s	-
34	6	60s	-
35	6	60s	-
36	6	60s	-
37	6	60s	-
38	6	60s	-
39	1	60s	2.58
40	1	60s	13.8
41	1	60s	48.5
42	1	60s	120.78
43	1	60s	307.6
44	2	60s	3.3

FIGURE 4.50: The Regression Data

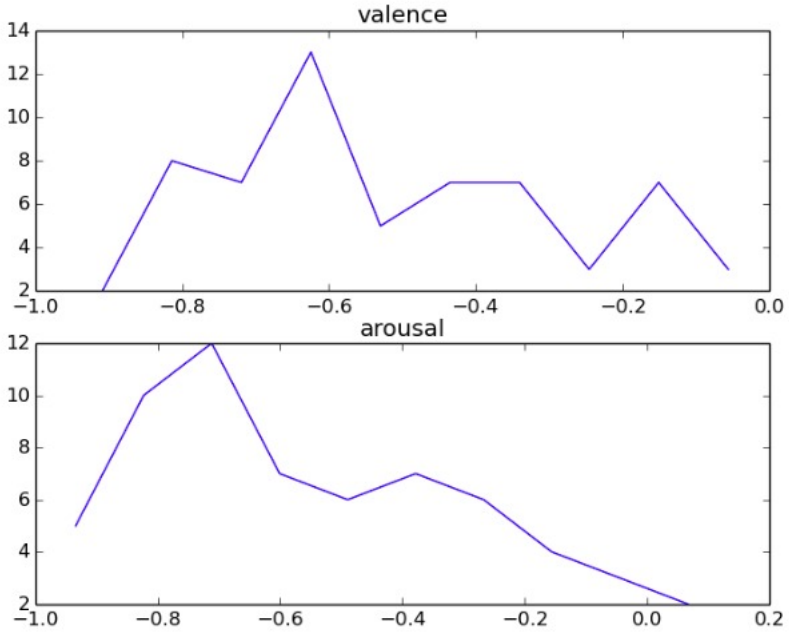


FIGURE 4.51: The Regression Result

The result was not accurate,because the data was recorded using different microphones which caused variation in the result.The solution was to use the Respeaker cause it the data will be recorded from the Respeaker itself,then we can train the data to achieve better results.

4.3.4 Final Implementation

4.3.5 Sensors

We used MRAA and UPM libraries to setup the sensors since all Raspberry Pi library like WiringPi was not able to work as long as the codes provided by Adfruit,we figure out that after reading the documentation provided for the board and it mention that to control the GPIO we must use MRAA and UPM [41].

We first installed MRRA and UPM using the following command:

```
sudo apt install  python-mraa python-upm libmraa1 libupm1 mraa-tools
```

4.3.6 BME280

Setting up the BME280 sensor was an easy task because it is supported in the UPM library.

- We used i2cdetect to scan an the I2C bus for devices by writing the command:

  ```
  sudo i2cdetect -y -r 3
  ```

  The result is I2C bus: 3 and Address: 77 as shown in Figure 4.52

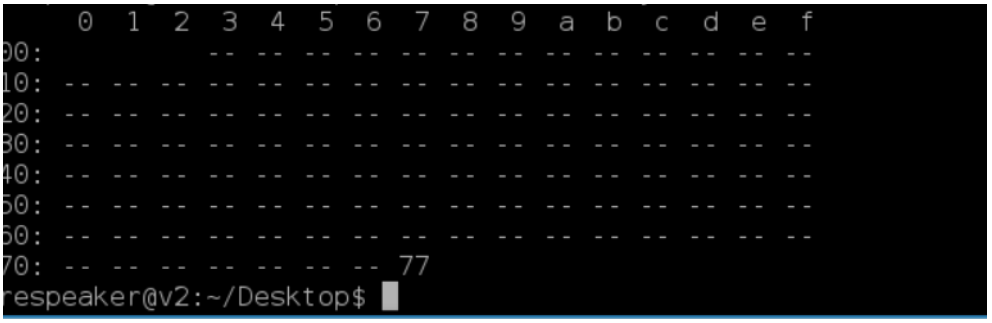


FIGURE 4.52: i2cdetect Scan Result for BME280

- Download the code from the upm library.
- Run the following code:

```
from __future__ import print_function
import time, sys, signal, atexit
from upm import pyupm_bmp280 as sensorObj

def main():
    # Instantiate a BME280 instance using default i2c bus and address
    sensor = sensorObj.BME280()

    # For SPI, bus 0, you would pass -1 as the address, and a valid pin for CS:
    # BME280(0, -1, 10);

    ## Exit handlers ##
    # This function stops python from printing a stacktrace when you hit control-C
    def SIGINTHandler(signum, frame):
        raise SystemExit

    # This function lets you run code on exit
    def exitHandler():
        print("Exiting")
        sys.exit(0)

    # Register exit handlers
    atexit.register(exitHandler)
    signal.signal(signal.SIGINT, SIGINTHandler)

    while (1):
        sensor.update()

        print("Compensation Temperature:", sensor.getTemperature(), "C /", end=' ')
        print(sensor.getTemperature(True), "F")

        print("Pressure: ", sensor.getPressure(), "Pa")

        print("Computed Altitude:", sensor.getAltitude(), "m")

        print("Humidity:", sensor.getHumidity(), "%RH")

        print()
        time.sleep(1)

if __name__ == '__main__':
    main()
```

- It worked as expected as shown in Figure 4.53

4.3.7 TSL2561

Even though the UPM library support the TSL2561 ,duo the different in the manufacturing company of the two chip, setting up the TSL2561 sensor was not as smooth as BME289 sensor.

```
Compensation Temperature: 27.7700004578 C / 81.986000061 F
Pressure:  100418.640625 Pa
Computed Altitude: 75.6908950806 m
Humidity: 35.3232421875 %RH
```

FIGURE 4.53: BME280 Results

- Scan the I2C bus to check if more devices where added by entering the following command:

```
sudo i2cdetect -y -r 3
```

The result is I2C bus: 3 and Address: 39 as shown in Figure 4.54

```
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- 39 -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- 77
respeaker@v2:~/Desktop$
```

FIGURE 4.54: i2cdetect Scan Result for TSL2561

- Download the code from the UPM library.
- Run the following code

```
from __future__ import print_function
import time, sys, signal, atexit
from upm import pyupm_tsl2561 as upmTsl2561

def main():
    # Instantiate a digital light sensor TSL2561 on I2C
    myDigitalLightSensor = upmTsl2561.TSL2561(0, 0x39)

    ## Exit handlers ##
    # This function stops python from printing a stacktrace when you hit control-C
    def SIGINTHandler(signum, frame):
        raise SystemExit

    # This function lets you run code on exit, including functions from
    myDigitalLightSensor
    def exitHandler():
        print("Exiting")
        sys.exit(0)

    # Register exit handlers
    atexit.register(exitHandler)
    signal.signal(signal.SIGINT, SIGINTHandler)

    while(1):
        print("Light value is " + str(myDigitalLightSensor.getLux()))
```

```
        time.sleep(1)

if __name__ == '__main__':
    main()
```

- But we got this error in Figure 4.55

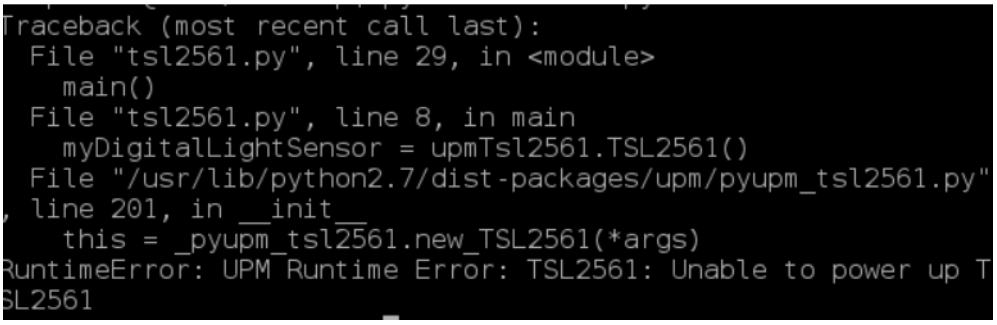


FIGURE 4.55: TSL2561 UPM Error

- After debugging the library ,we found that the address register for the device was 29 which does not match with what we got from i2cdetect, so in the main file we passed the function the correct address (39) by edit this line in the main file by :

```
myDigitalLightSensor = upmTsl2561.TSL2561(0, 0x39)
```

- After this change every things worked fine as shown in Figure 4.56

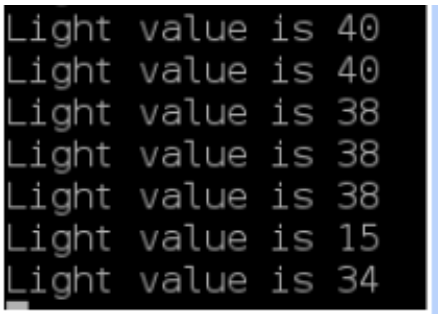


FIGURE 4.56: TSL2561 Results

4.3.8 LSM9DS0

We faced a lot of difficulties setup this sensor,duo the fact that it is not supported by UPM library.

- First, we tried to to use the UPM code for the LSM9DS0 which is an older version of the sensor, but unfortunately we keep getting errors indicating that it was not able to connect to the sensor I2C address as shown in Figure 4.57

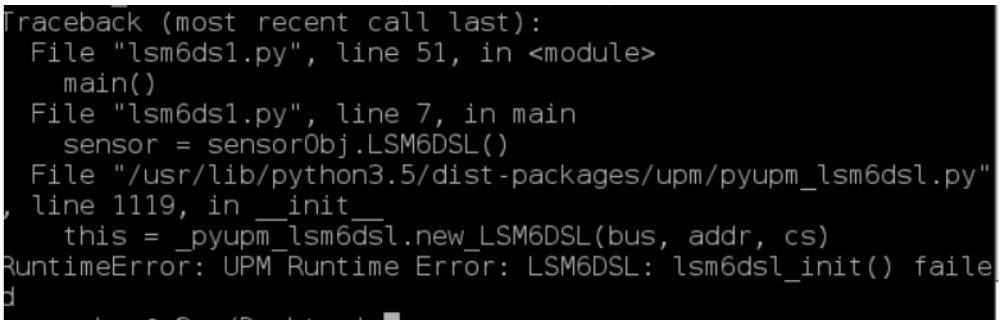


FIGURE 4.57: LSM9DS0 Error

So the only solution was to write the code using MRRA library.

- Second, we used i2cdetact to find the sensor address as shown in Figure 4.58 and we found that the address is 57.
- So we changed the address from 1D to 57, but we faced the same problem.

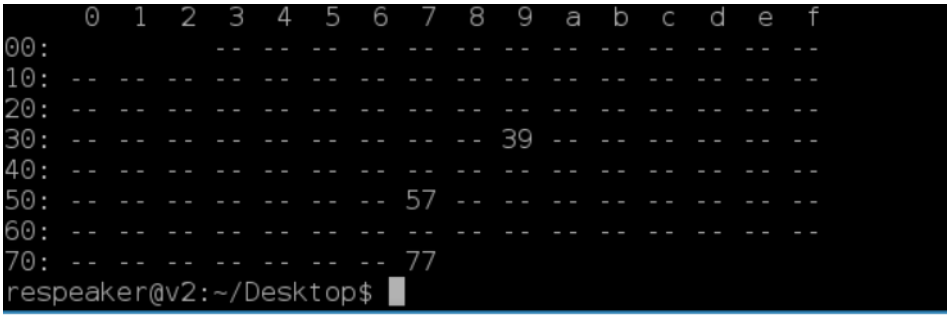


FIGURE 4.58: i2cdetact for LSM9DSO

- We found out that the sensor have two address one for the Accelerometer and gyroscope(0x57) and the other Magnetic sensor(0x1E) as shown in C. But also that did not solve the problem.
- We compared both sensor data sheet and found that register address for both sensor was totally different as shown in C
- At this point we decided to get help and ask people on StackOverFlow as shown in Figure 4.59
- The answer was you have to create your own library using MRRA.
- As last solution we decided to try to create a new library for the LSM9DS1 using MRAA.We used the `Adafruit_LSM9DS1` library that is used for raspberry pi to get the all the configuration and registers address.

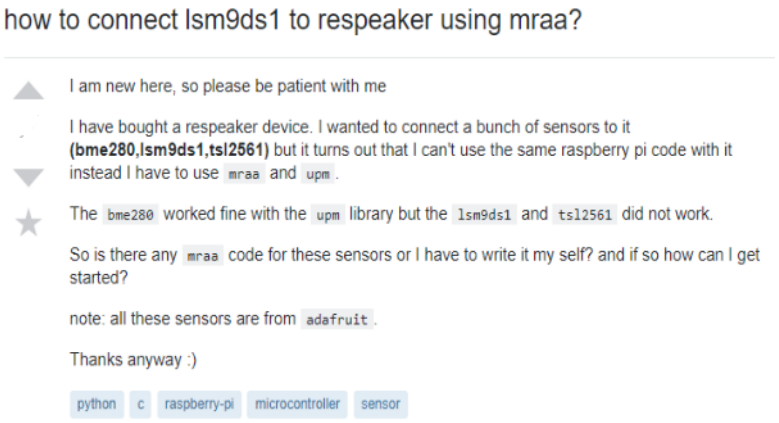


FIGURE 4.59: StackOverFlow Question

```
class XM:
 ADDRESS = 0x6B
 WHO_AM_I = 0x0F     # r
 WHO_AM_I_OK = 0x68  #r
 ACT_THS = 0x04 #rw
 ACT_DUR = 0x05 # rw
 INT_GEN_CFG_XL = 0x06 # rw
 INT_GEN_THS_X_XL = 0x07 # rw
 INT_GEN_THS_Y_XL = 0x08 # rw
 INT_GEN_THS_Z_XL = 0x09 # rw
 INT_GEN_DUR_XL = 0x0A # rw
 REFERENCE_G = 0x0B # rw
 INT1_CTRL = 0x0C # rw
 INT2_CTRL = 0x0D # rw
 CTRL_REG1_G = 0x10 #rw
 CTRL_REG2_G = 0x11 #rw
 CTRL_REG3_G = 0x12 # rw
 ORIENT_CFG_G = 0x13 # rw
 INT_GEN_SRC_G = 0x14 # r
 OUT_TEMP_L = 0x15 # r
 OUT_TEMP_H = 0x16 # r
 STATUS_REG = 0x17 # r
 OUT_X_L_G = 0x18 # r
 OUT_X_H_G = 0x19 # r
 OUT_Y_L_G = 0x1A # r
 OUT_Y_H_G = 0x1B # r
 OUT_Z_L_G = 0x1C # r
 OUT_Z_H_G = 0x1D # r
 CTRL_REG4 = 0x1E # rw
 CTRL_REG5_XL = 0x1F # rw
 CTRL_REG6_XL = 0x20 # rw
 CTRL_REG7_XL = 0x21 # rw
 CTRL_REG8 = 0x22 # rw
 CTRL_REG9 = 0x23 # rw
 CTRL_REG10 = 0x24 # rw
 Reserved = 0x25 # none
 INT_GEN_SRC_XL = 0x26 # r
```

```
        STATUS_REG = 0x27 # r
        OUT_X_L_XL = 0x28 # r
        OUT_X_H_XL = 0x29 # r
        OUT_Y_L_XL = 0x2A # r
        OUT_Y_H_XL = 0x2B # r
        OUT_Z_L_XL = 0x2C # r
        OUT_Z_H_XL = 0x2D # r
        FIFO_CTRL = 0x2E # rw
        FIFO_SRC = 0x2F # r
        INT_GEN_CFG_G = 0x30 # rw
        INT_GEN_THS_XH_G = 0x31 # rw
        INT_GEN_THS_XL_G = 0x32 # rw
        INT_GEN_THS_YH_G = 0x33 # rw
        INT_GEN_THS_YL_G = 0x34 # rw
        INT_GEN_THS_ZH_G = 0x35 # rw
        INT_GEN_THS_ZL_G = 0x36 # rw
        INT_GEN_DUR_G =  0x37 # rw
        RANGE_A = {'2G':(0b00 << 3),'4G':(0b10 << 3),'8G':(0b11 << 3),'16G':(0b01 <<
        3)}
        CAL_A = {'2G': (2.0/32768.0),'4G': (4.0/32768.0),'8G':
        (8.0/32768.0),'16G':(16.0/32768.0)}
        CAL_TEMP = 1.0/8.0
        RANGE_G = {'245DPS': (0b00 << 3), '500DPS': (0b01 << 3), '2000DPS': (0b11 <<
        3)}
        CAL_G = {'245DPS': (245.0/32768.0), '500DPS': (500.0/32768.0), '2000DPS':
        (2000.0/32768.0)}
    # Class for gyro configuration and registers
    class MAG:
        ADDRESS = 0x1E
        WHO_AM_I = 0x0F # r
        WHO_AM_I_OK = 0x3D #r
        OFFSET_X_REG_L_M = 0x05 # rw
        OFFSET_X_REG_H_M = 0x06 # rw
        OFFSET_Y_REG_L_M = 0x07 # rw
        OFFSET_Y_REG_H_M = 0x08 # rw
        OFFSET_Z_REG_L_M = 0x09 # rw
        OFFSET_Z_REG_H_M = 0x0A # rw
        CTRL_REG1_M = 0x20 # rw
        CTRL_REG2_M = 0x21 # rw
        CTRL_REG3_M = 0x22 # rw
        CTRL_REG4_M = 0x23 # rw
        CTRL_REG5_M = 0x24 # rw
        STATUS_REG_M = 0x27 # r
        OUT_X_L_M = 0x28 # r
        OUT_X_H_M = 0x29 # r
        OUT_Y_L_M = 0x2A # r
        OUT_Y_H_M = 0x2B # r
        OUT_Z_L_M = 0x2C # r
        OUT_Z_H_M = 0x2D # r
        INT_CFG_M = 0x30 # rw
        INT_SRC_M = 0x31 # r
        INT_TSH_L_M = 0x32 # rw
        INT_TSH_H_M = 0x33 # rw
        RANGE_M = {'4GAUSS': (0b00 << 5), '8GAUSS': (0b01 << 5), '12GAUSS': (0b10 <<
        5), '16GAUSS': (0b11 << 5)}
```

```
    CAL_M = {'4GAUSS': (2.0/32768.0), '8GAUSS': (4.0/32768.0), '12GAUSS':
    (8.0/32768.0), '16GAUSS': (12.0/32768.0)}
```

We used the LSM9DS0 MRAA library to implement all the library functions.The link for the library is here [42]

- After running the following code:

```
from LSM9DS1 import IMU
import time
# Create IMU object
imu = IMU() # To select a specific I2C port, use IMU(n). Default is 1.
# Initialize IMU
imu.initialize()
# Enable accel, mag, gyro
imu.enable_accel()
imu.enable_mag()
imu.enable_gyro()
# Set range on accel, mag, and gyro
# Specify Options: "2G", "4G", "6G", "8G", "16G"
imu.accel_range("2G")       # leave blank for default of "2G"
# Specify Options: "2GAUSS", "4GAUSS", "8GAUSS", "12GAUSS"
imu.mag_range("2GAUSS")     # leave blank for default of "2GAUSS"
# Specify Options: "245DPS", "500DPS", "2000DPS"
imu.gyro_range("245DPS")    # leave blank for default of "245DPS"
# Loop and read accel, mag, and gyro
while(1):
    imu.read_accel()
    imu.read_mag()
    imu.read_gyro()

    # Print the results
    print "Accel: " + str(imu.ax) + ", " + str(imu.ay) + ", " + str(imu.az)
    print "Mag: " + str(imu.mx) + ", " + str(imu.my) + ", " + str(imu.mz)
    print "Gyro: " + str(imu.gx) + ", " + str(imu.gy) + ", " + str(imu.gz)
```

- it works as shown in Figure 4.60

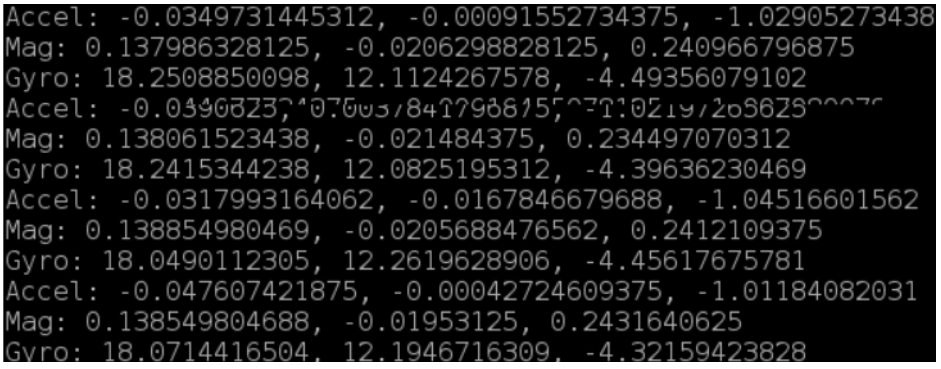


FIGURE 4.60: LSM9DS0 Results

4.3.9 Skill

The Skill help the user to interact with the voice assistance,So we developed a Skill to know all the possible narration flows of the conversation between the user and the voice assistance.

4.3.10 Alexa Skills Kit

We have to use Alexa Skills kit to specify the invocation,for more information about Alexa Skills Kit refer to 3.

1. The first step is to create a new skill for Super Sensor in the developer console.

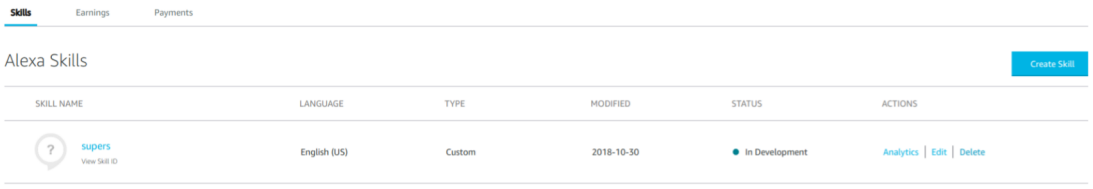


FIGURE 4.61: Create Skill for Super Sensor

2. Next, we set up our skill's invocation name.

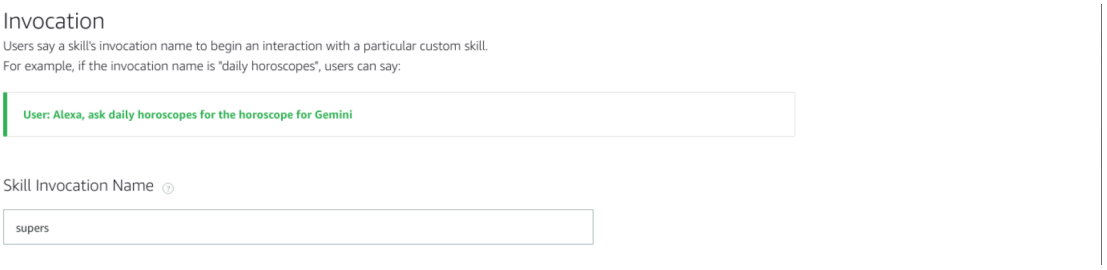


FIGURE 4.62: Super Sensor Invocation Name

3. New step, we create a new intent for the question of room description where all different grammatical forms of the same question will be stored so that it can pick up the request of description from the user later on.

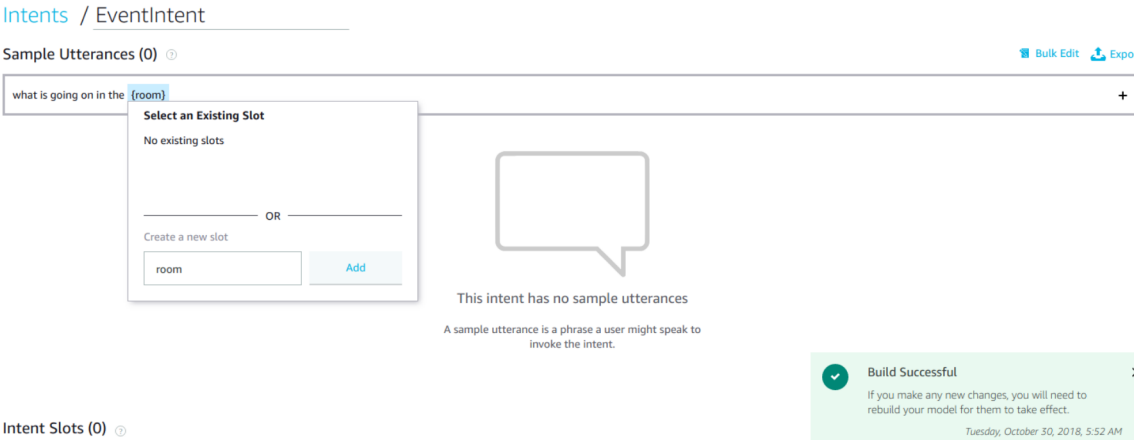


FIGURE 4.63: Describe Intent

4. Finally, we specify the endpoint for our skill

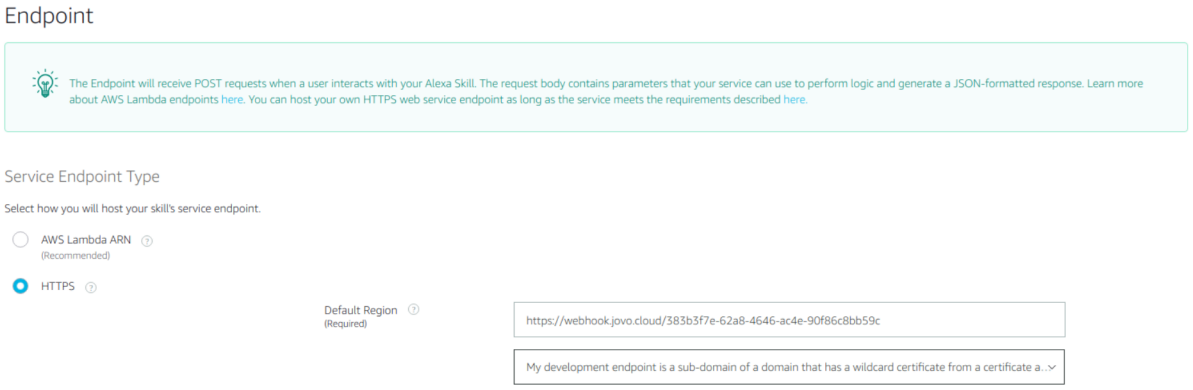


FIGURE 4.64: HTTP Endpoint

4.3.11 Jovo Endpoint

In this part, we will implement the Jovo endpoint for our Super Sensor skill, which means that we will write a code that reply to each request that comes from the user.

- Create a new project

```
jovo new supers
```

This will give us the default skill code from Amazon.

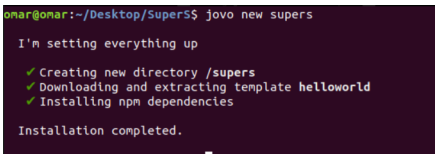


FIGURE 4.65: Create a new project using jovo

- Generate the Alexa skill JSON file

```
jovo init alexaSkill
```

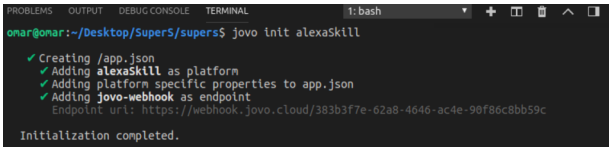


FIGURE 4.66: Jovo init

- Build the project

```
jovo build
```

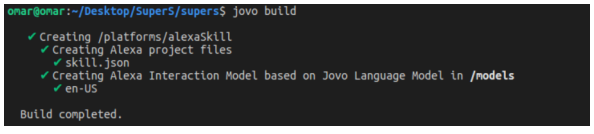


FIGURE 4.67: Build the project

- Deploy the project

```
jovo deploy
```

This will connect the Alexa developer kit that we created in the previous section to our jovo skill.

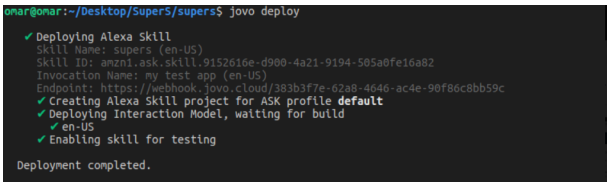


FIGURE 4.68: Deploy the project

- We should install Firebase by:

```
npm install firbase-admin-sdk
```

- Finally, we run the following jovo code for this Skill:

```
%First, We need to setup our Firebase connection.%
'use strict';


var admin = require("firebase-admin");

var serviceAccount = require("/home/omar/Desktop/SuperS/supers/supers-87d35-firebase
-adminsdk-oizo3-454ae9147b.json");

admin.initializeApp({
  credential: admin.credential.cert(serviceAccount),
  databaseURL: "https://supers-87d35.firebaseio.com"
});
var room="Al yasat hall"
var db = admin.database();
%Setup our Jovo App configuration%
// ================================================================================
// App Configuration
// ================================================================================

const {App} = require('jovo-framework');

const config = {
    logging: false,
};


const app = new App(config);

%LAUNCH Function%
// ================================================================================
// App Logic
// ================================================================================

app.setHandler({
```

```
    'LAUNCH': function() {
        this.ask('<voice name="Brian"><lang xml:lang="en-GB">Hallo sir, I am
        alfred your home assistants,how may i help you today?.</lang></voice>');
    },

    'EventsIntent': async function() {
        var ref = db.ref('/'+room+"/events");
        var event=await ref.once("value").then((snapshot) =>{
            return snapshot.val();});
        this.tell('<voice name="Brian"><lang
        xml:lang="en-GB">'+event.toString()+'</lang></voice>');
    },
    'TemperatureIntent':async function() {
        var ref = db.ref('/'+room+"/temperature");
        var temperature=await ref.once("value").then((snapshot) =>{
            return snapshot.val();});
            this.tell('<voice name="Brian"><lang xml:lang="en-GB">
            the temperature right now in the
            '+room+', is '+temperature+"
            Celsius</lang></voice>");
    },
    'lightintent':async function() {
        var ref = db.ref('/'+room+"/light");
        var light=await ref.once("value").then((snapshot) =>{
            return snapshot.val();});
            this.tell('<voice name="Brian"><lang xml:lang="en-GB">the light
            right now in the '+room+', is '+light+"</lang></voice>");
    },
    'EventIntent':async function() {
        var ref = db.ref('/'+room+"/cevent");
        var cevent=await ref.once("value").then((snapshot) =>{
            return snapshot.val();});
            this.tell('<voice name="Brian"><lang xml:lang="en-GB">right now I
            can hear '+cevent+ " in the "+room+"</lang></voice>");
    },
    'consumptionIntent':async function() {
        var ref = db.ref('/'+room+"/waterconsumption");
        var waterconsumption=await ref.once("value").then((snapshot) =>{
            return snapshot.val();});
            this.tell('<voice name="Brian"><lang xml:lang="en-GB">
            The total water consumption for the month of november so far is
            '+waterconsumption+" liters</lang></voice>");
    },
});

module.exports.app = app;
```

- Enter the following command to run the Skill:

```
    jovo run
```

- As shown in the Figure below, the Skill is running and you can now talk to Alexa!!

FIGURE 4.69: Jovo Skill running

- The code has five services:

1. The first service, it list all the previous events.The code for it is:

```
'EventsIntent': async function() {
    var ref = db.ref('/'+room+"/events");
    var event=await ref.once("value").then((snapshot) =>{
        return snapshot.val();});
    this.tell('<voice name="Brian"><lang
    xml:lang="en-GB">'+event.toString()+'</lang></voice>');
},
```

2. The second service,it give you the current temperature in the room.The code for it is:

```
'TemperatureIntent':async function() {
    var ref = db.ref('/'+room+"/temperature");
    var temperature=await ref.once("value").then((snapshot) =>{
        return snapshot.val();});
        this.tell('<voice name="Brian"><lang xml:lang="en-GB">
        the temperature right now in the '+room+', is '+temperature+"
        Celsius</lang></voice>");
},
```

3. The third service ,The light status at the room.The code for it is:

```
'lightintent':async function() {
    var ref = db.ref('/'+room+"/light");
    var light=await ref.once("value").then((snapshot) =>{
        return snapshot.val();});
        this.tell('<voice name="Brian"><lang xml:lang="en-GB">the light
        right now in the '+room+', is '+light+"</lang></voice>");
},
```

4. The forth service, it list all the current event.The code for it is:

```
'EventIntent':async function() {
    var ref = db.ref('/'+room+"/cevent");
    var cevent=await ref.once("value").then((snapshot) =>{
        return snapshot.val();});
```

```
        this.tell('<voice name="Brian"><lang xml:lang="en-GB">right now I
        can hear '+cevent+ " in the "+room+"</lang></voice>");
},
```

5. The fifth service,the water consumption for the month.The code for it is:

```
'consumptionIntent':async function() {
    var ref = db.ref('/'+room+"/waterconsumption");
    var waterconsumption=await ref.once("value").then((snapshot) =>{
        return snapshot.val();});
        this.tell('<voice name="Brian"><lang xml:lang="en-GB">
        The total water consumption for the month of november so far is
        '+waterconsumption+" liters</lang></voice>");
},
```

6. you can change the services above based on what you need.

Chapter 5

Project Management

5.1 Team Members

In this section we will introduce each team member, and proceed to explain the time-line and the progression of the project.

5.1.1 Omar Al Tawil

For our first member, who is also our team leader Omar Al Tawil. Omar is a hard working student who is currently pursuing bachelors in Computer Engineering , and the glue of this group, he kept the group going to finish the project as early as possible so that we could begin with the testing stages, so that we perfect the project. He is the best debugger of the group and helped fix many errors in the codes, plus he is a great researcher. One of the weak points in Tawil is designing, plus since he is from an Arabic education background his English writing skills are not that good.

5.1.2 Ahmad Hashi

For our second member, Ahmad a Computer Engineer is the team manager, he dealt with time management and meetings, that is why our project management was organized and done properly. He helped with the design aspects and the hardware circuits, plus Ahmad helped organize our weekly meetings and to divide the work on all of us.

5.1.3 Mahmoud Bakir

For our third member, Mahmoud a Computer Engineer and he has a great programming skills, and an expert in electric circuits.Mahmoud helps in the hardware part in this project as he is the one who designed most of the circuits with their PCB(Printed Board Circuits).He also worked on the coding of the project.Mahmoud is always available for every group meeting,and his weaknesses are in soldering .

5.1.4 omar Aoun

The other Omar in the group is a passionate student, who is also a Computer Engineer. Omar had a passion for computers since he was young and that is why he went for the bachelors in Computer Engineering, he also is a good designer who designed the 3D designs, and also worked on the software part of the project.

5.2 Sharing Hubs

In our journey, we have used two main hubs to share files and folders and keeps them saved, files from codes to circuit designs to research paper etc.. and these two hubs are:

5.2.1 Google Drive

We have used Google drive because it is easy to use and very well organized. We saved all of the things that we have used in our project or will use for future aspects.

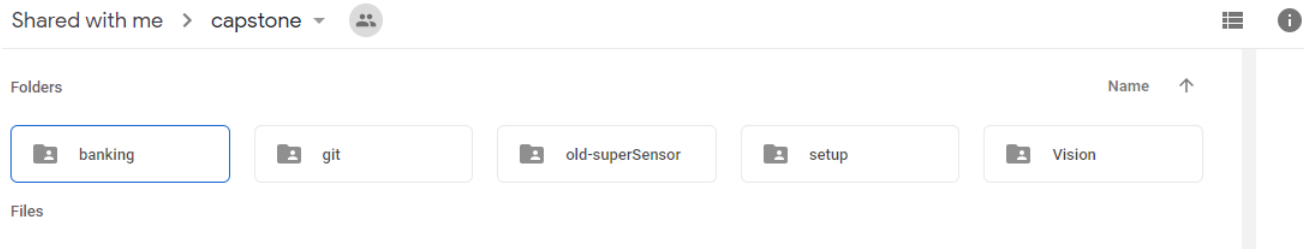


FIGURE 5.1: drive

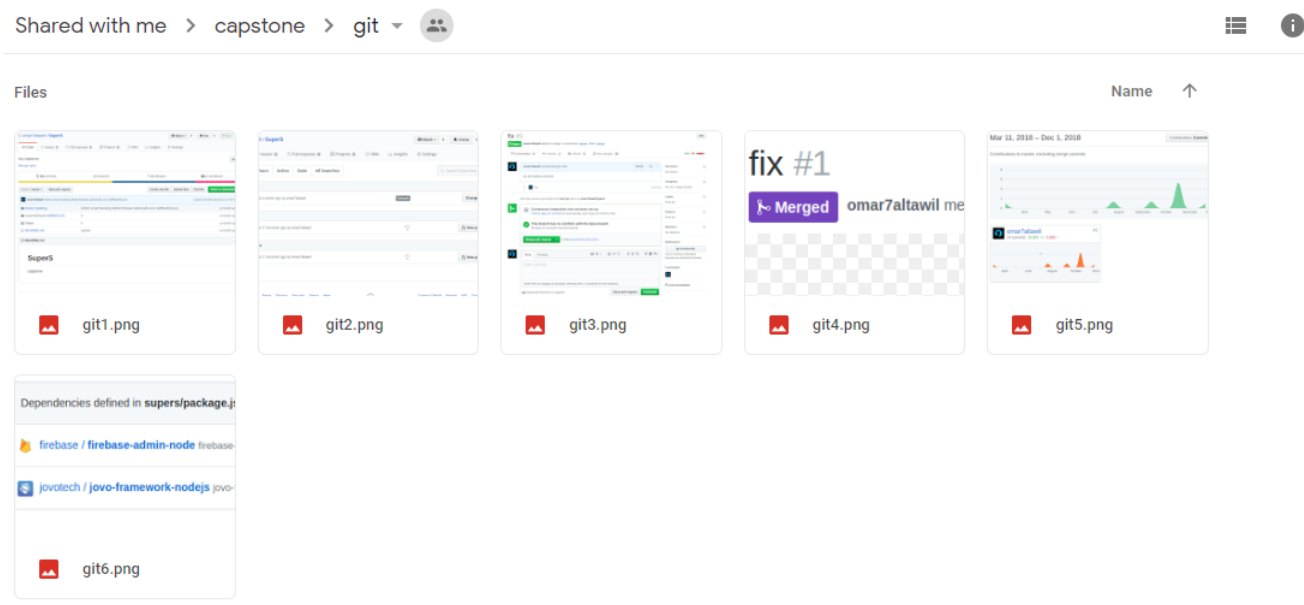


FIGURE 5.2: drive

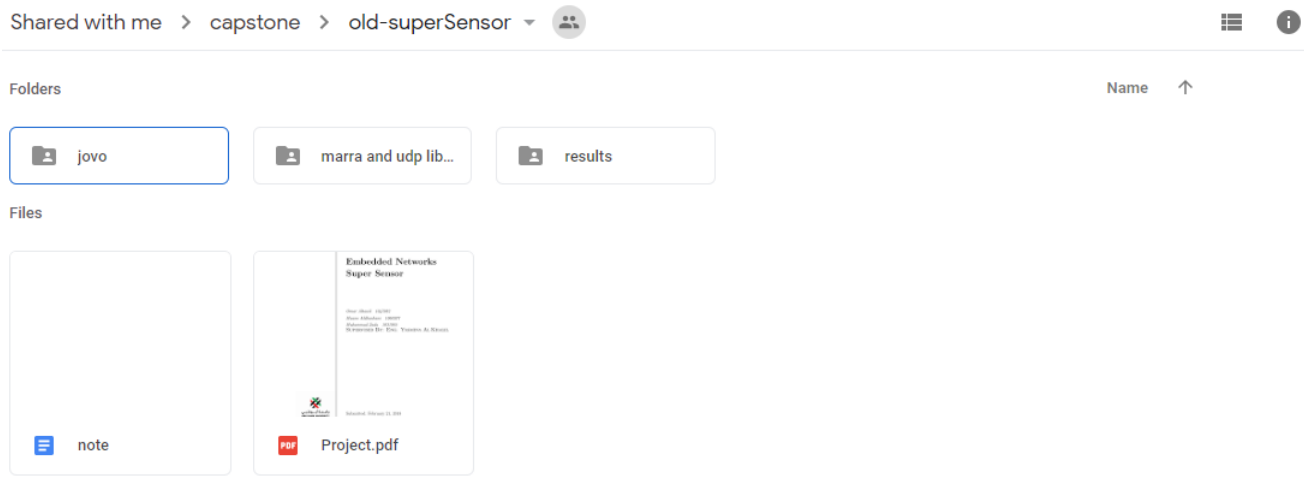


FIGURE 5.3: drive

5.2.2 Github

The other storage environment is GitHub. GitHub is used by all programmers around the world to store codes and share them, and that is where we kept all of our codes.

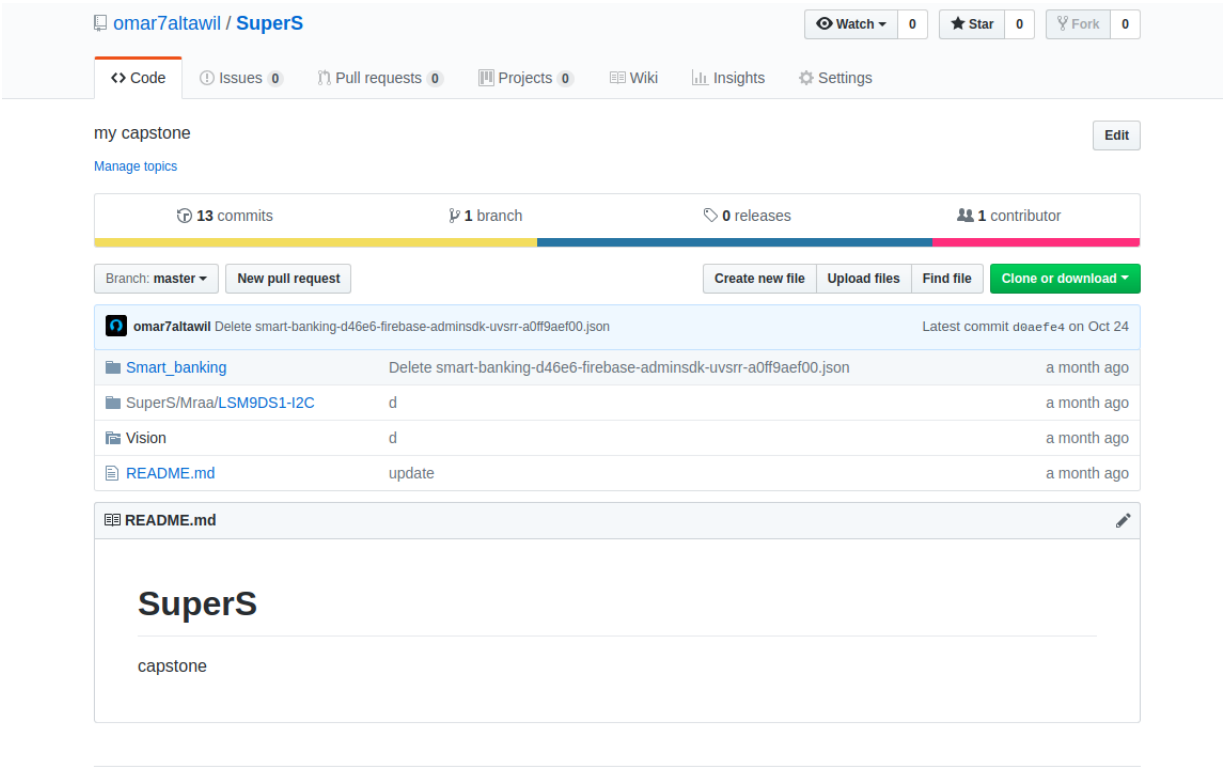


FIGURE 5.4: GitHub

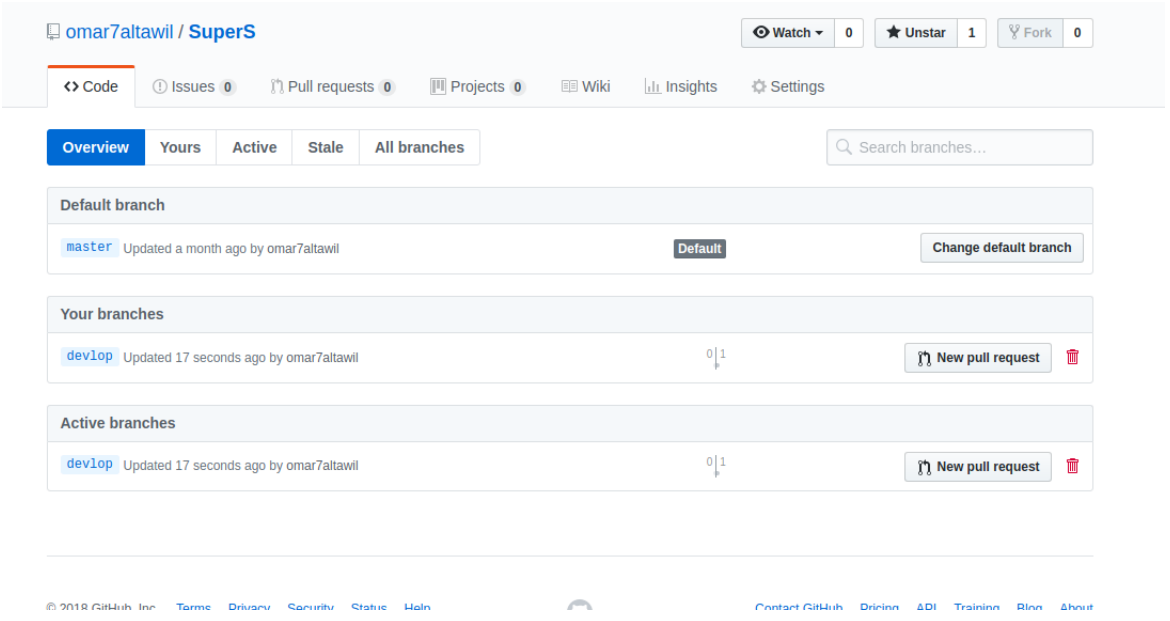


FIGURE 5.5: GitHub

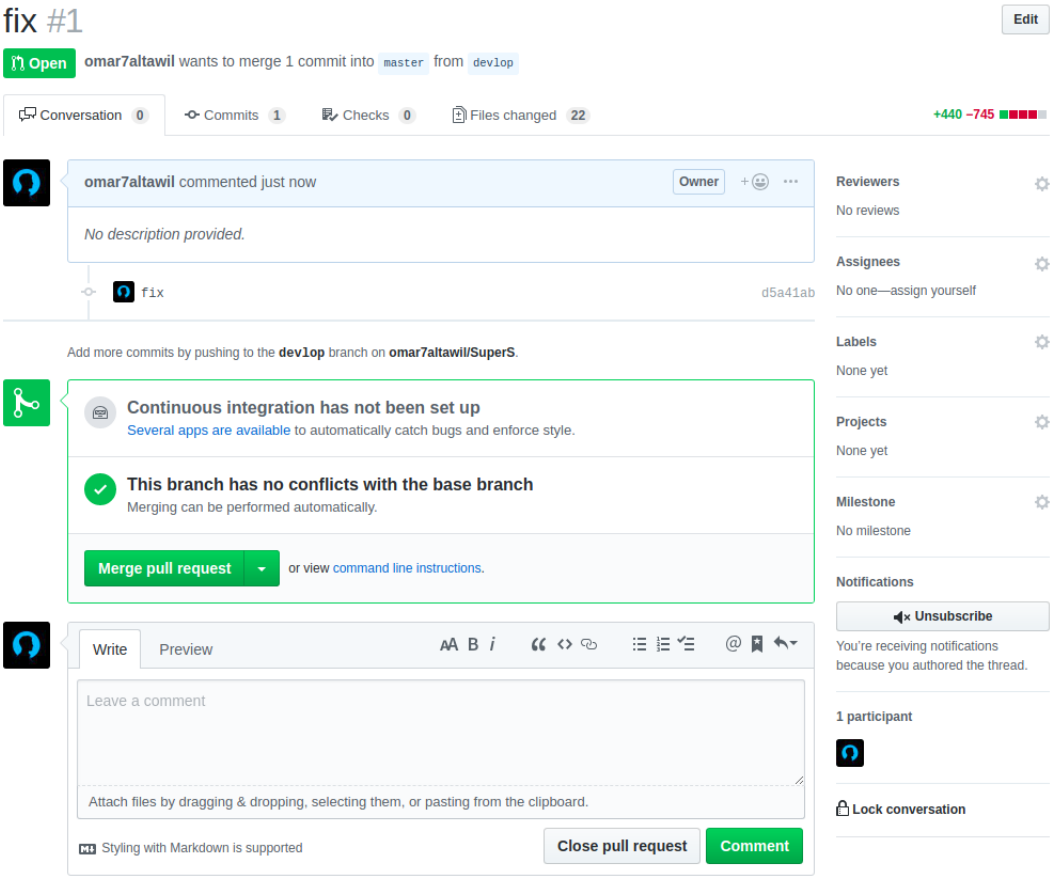


FIGURE 5.6: GitHub

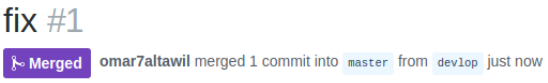


FIGURE 5.7: GitHub

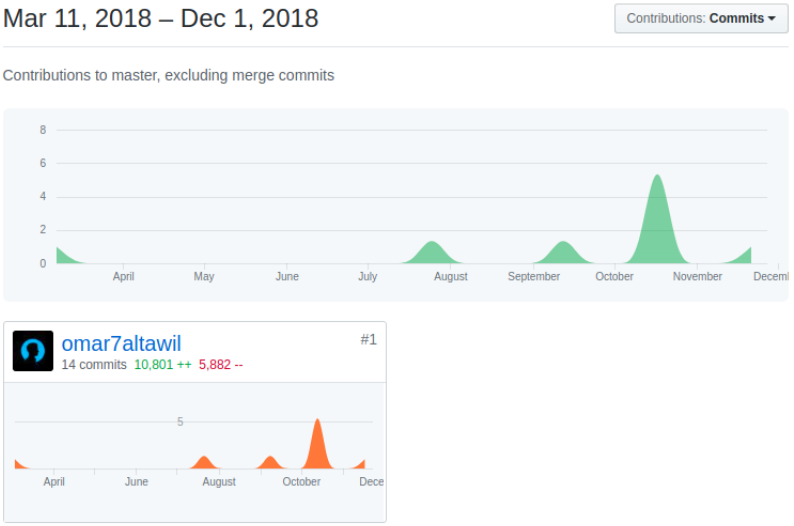


FIGURE 5.8: GitHub

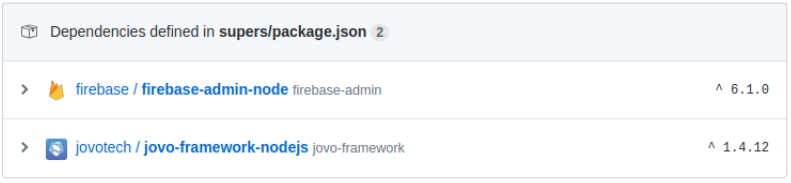


FIGURE 5.9: GitHub

5.3 Tasks

In this section, we are going to explain the tasks that are in this project, together with their time stamps. A table containing all of the tasks with their start and end dates is found below:

Tasks	Start Date	End Date
Super sensor 1:circut	Dec-17	Jan-18
Super sensor 1: Android App	Dec-17	Jan-18
Super sensor 1: Adruino	Dec-17	Jan-18
Super sensor 1: Python	Dec-17	Jan-18
Super sensor 1: Collecting data Phase 1	Feb-18	Apr-18
Super sensor 1: Training and Testing Phase 1	Apr-18	Apr-18
Super sensor 1: Collecting data Phase 2	May-18	May-18
Super sensor 1: Training and Testing Phase 2	May-18	May-18
Super sensor 2:circut	Aug-18	Sep-18
Super sensor 2:Maraa and UDP	Sep-18	Oct-18
Alexa setup	Sep-18	Oct-18
Smart Banking: Button Circuit	Oct-18	Oct-18
Smart Banking: Button code	Oct-18	Nov-18
Smart Banking: Skill	Oct-18	Nov-18
Smart Banking: PCB	Oct-18	Nov-18
Smart Banking: 3D Design	Oct-18	Nov-18
Vision:Hardware setup	Nov-18	Nov-18
Vision:Code	Nov-18	Nov-18
Vision:Skill	Nov-18	Nov-18
Super sensor 2:Skill	Nov-18	Dec-18
respaker PCB	Dec-18	Dec-18

Table 5.1: Tasks Table

5.4 Components Prices

Item	Quantity	Total Price(AED)
VGA to HDMI converter	1	57
PS2 Keyboard/Mouse	1	39
Ultrathin Heat Sink	1	6
Respeaker V2	2	754
SD Card	1	151
SD Card Reader	1	40
Echo Dot	1	180
ESP01	1	10
Raspberry PI sense Hat	1	147
Total		1384

Table 5.2: Item Prices

5.5 Communication

Communication is a critical part of our project, without it the project would have failed and we wouldn't be able to do anything. Communication helped us to work as a team and divide the work upon us equally. We have used many ways of communicating, some are:

5.5.1 Emails

Emails is the most traditional way to communicate with other colleagues, we used emails to setup meetings with Dr. Mohammad Ghazal by sending calendar events, and we also used them to send each other the files that are needed to continue the work.

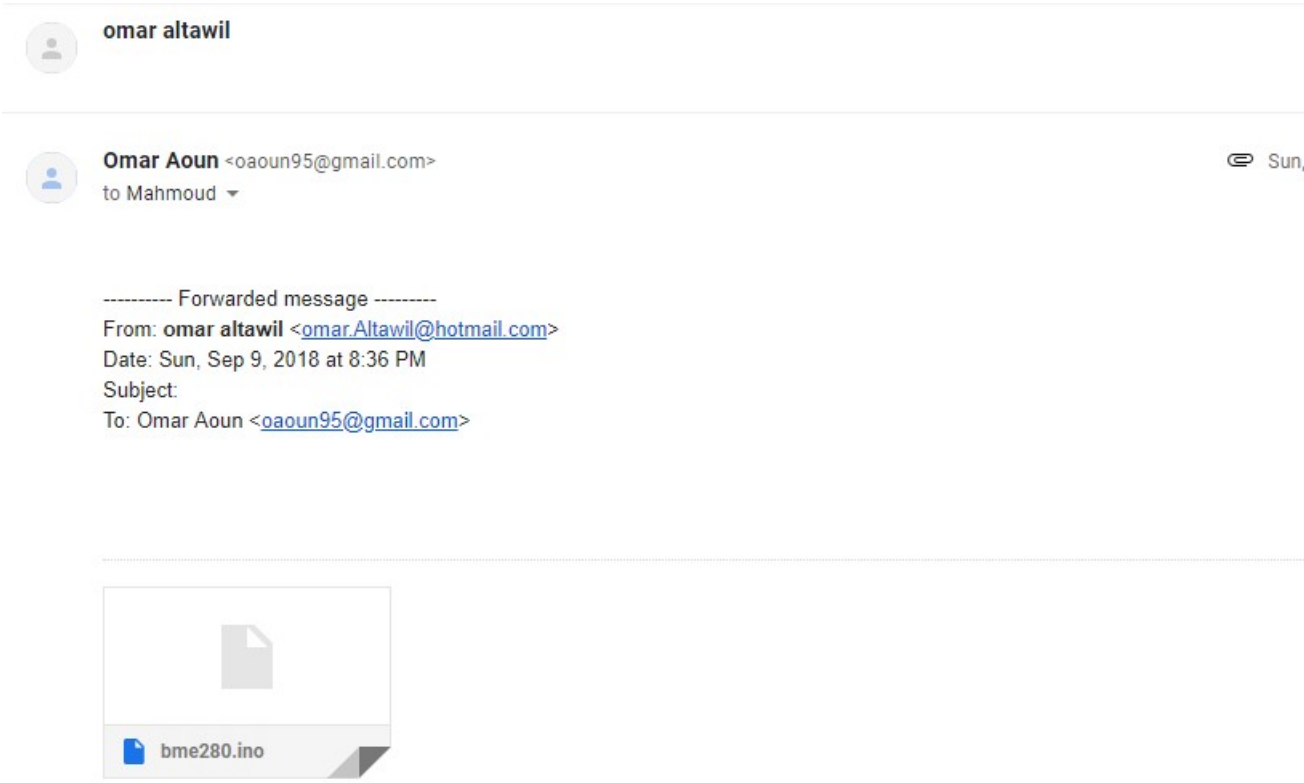
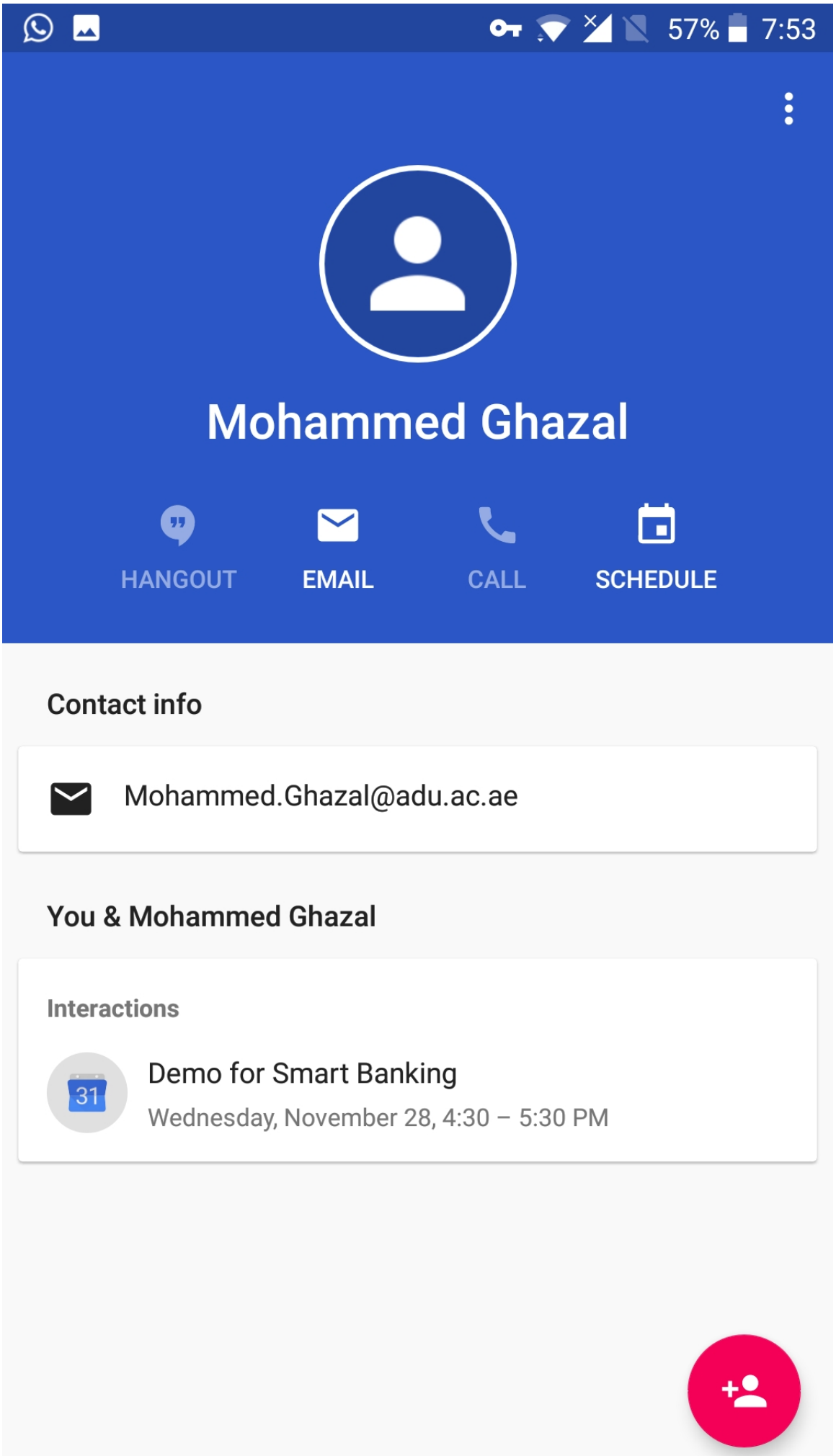


FIGURE 5.10: email

FIGURE 5.11: calendar event1


57% 7:53
Mohammed Ghazal
HANGOUT
EMAIL
CALL
SCHEDULE
Contact info
Mohammed.Ghazal@adu.ac.ae
You & Mohammed Ghazal
Interactions
31
Demo for Smart Banking
Wednesday, November 28, 4:30 – 5:30 PM

5.5.2 Meetings

Another type of communications that we used was our weekly meetings, we used to agree on the weekly meetings using either emails, or whatsapp, or by confirm face to face on when the meeting will take place and where.

5.5.3 Whatsapp

One of the most used method of communication that we used was on whatsapp, we used whatsapp to set a meeting, send files, talk about problems in the project, and manage the time and the work of everything.

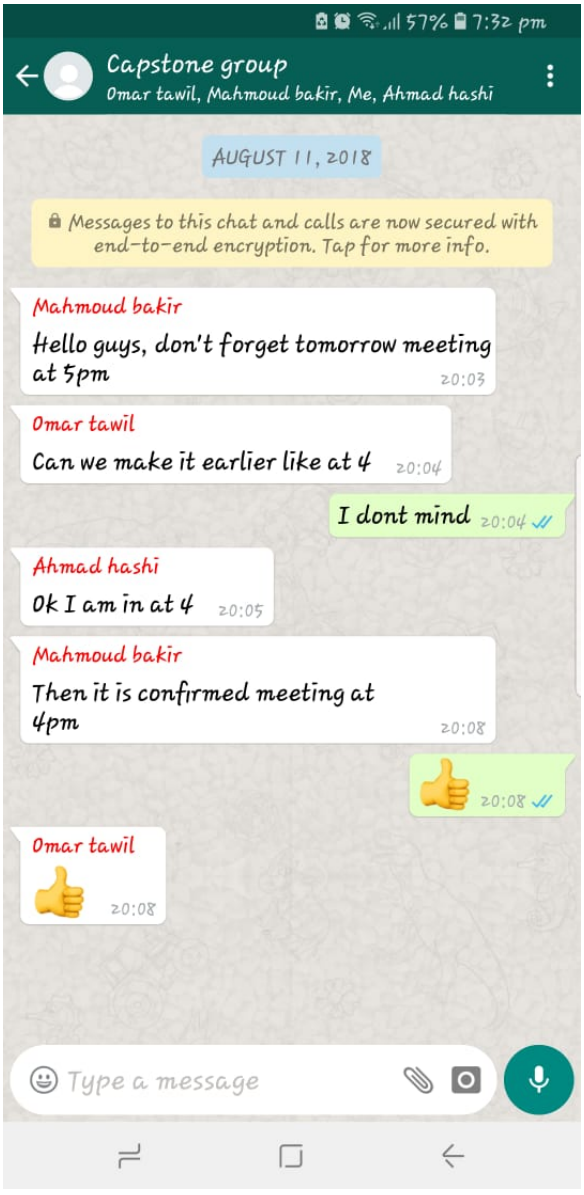


FIGURE 5.13: whatsapp1

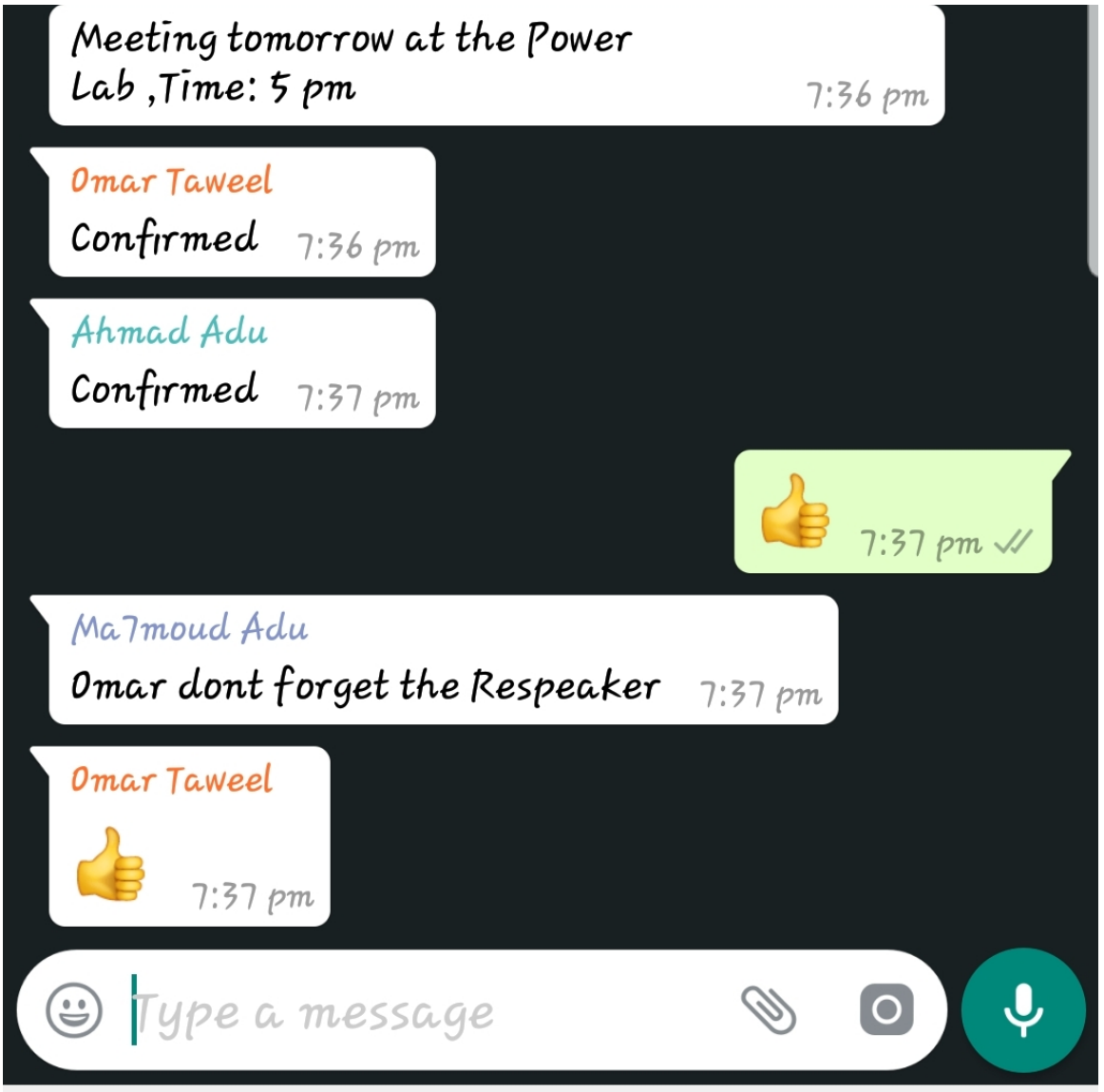


FIGURE 5.14: whatsapp2

Chapter 6

Results and Discussion

After we have deployed our system, we tested all the three skills that were implemented within the system. The testing results for each skill is shown downward.

6.1 Testing Skill: Vision

Requirements Tested:- Angular Application, Description Service, Count People Service, Google Vision API.

1. We start by opening the Angular page
2. Next, we add the description of the room
3. Then, we add the name of the room
4. Now, from the Respeaker we run the Camera code which contains Vision API
5. Next, we run the Jovo skill and Endpoint and ask Alexa about a given service:
   (a) Description Service
   (b) Count People Service

Expected Results:

1. The Angular Application boots up with all of its functions working
   (a) Add A New Room

(b) Delete An Existing Room

(c) Edit An Existing Room

2. All of the information should be uploaded to the Real-time Database thought Firebase

3. The Python code that runs the Vision API should extract labels from the images that are generated using the Camera

4. Vision API also gives us the count of how many persons are in the room through face detection

5. When the user invoke the name of the skill when Alexa is running, it should respond

   (a) When the user invoke the description service, it should combine the description from Angular Application and labels from Vision API

   (b) The Count service should return how many people in the room

Actual Results:

1. The Angular Application boots up and functions perfectly

   (a) Adding A New Room function is working

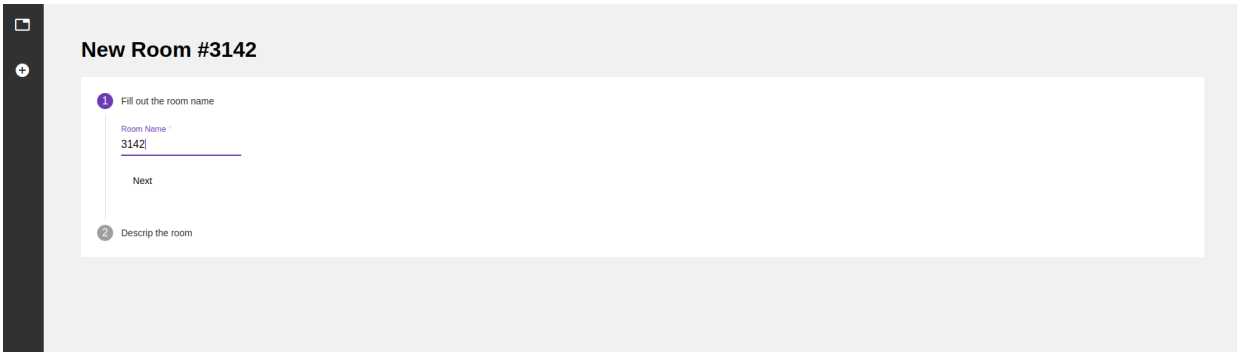


FIGURE 6.1: New Room: Name

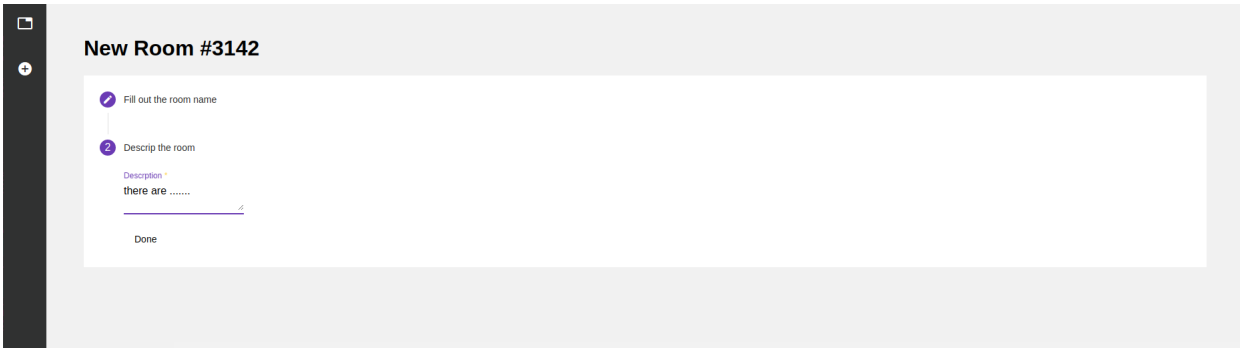


FIGURE 6.2: New Room: Description

(b) Deleting An Existing Room function is working

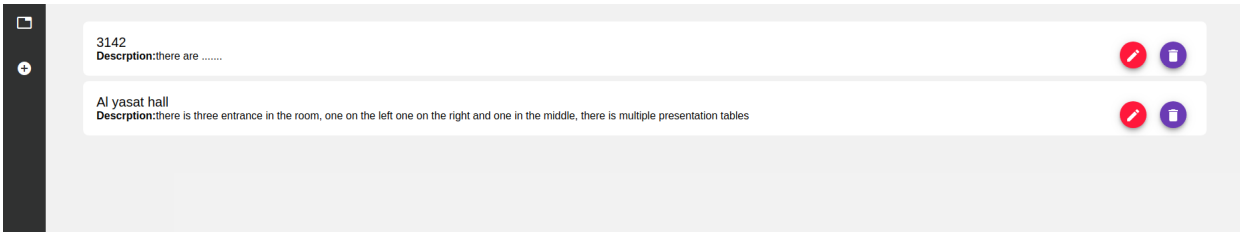


FIGURE 6.3: Deleting A Room

(c) Editing An Existing Room function is working

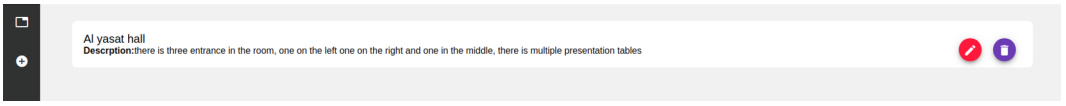


FIGURE 6.4: Editing A Room

2. The connection between the Angular Application and Real-time Database is functioning

FIGURE 6.5: Firebase Connection

3. Using the images taken from the camera, the Python code successfully extracts labels

FIGURE 6.6: Extracting Labels

4. The function of Vision API gives us the count of how many persons are in the room through face detection is working

5. When the user invoke the name of the skill when Alexa is running, it responds to the user

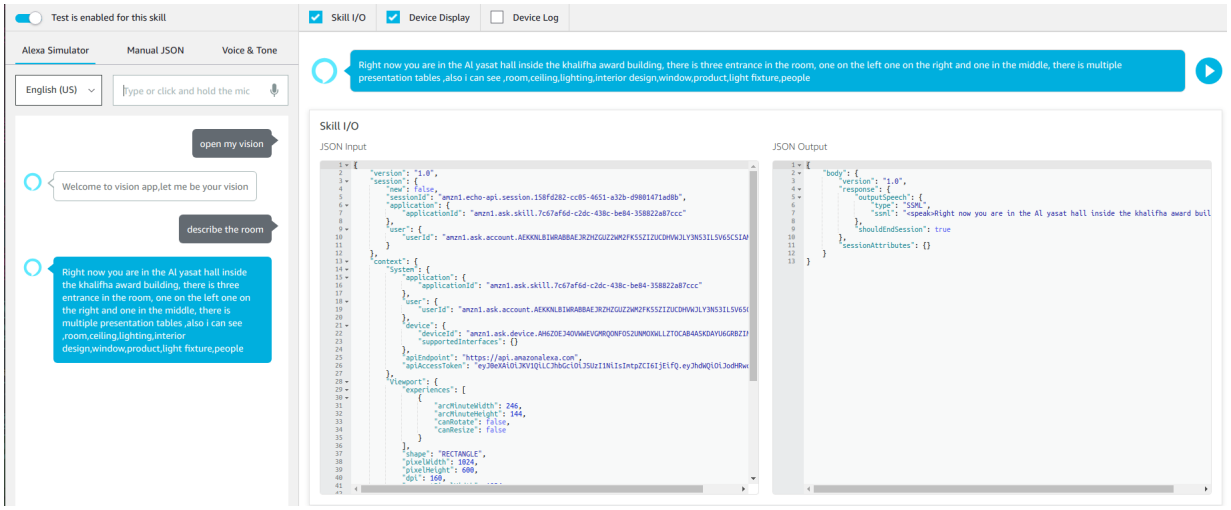


FIGURE 6.7: Alexa Response

(a) When the user invoke the description service, it combines the description from Angular Application and labels from Vision API successfully

(b) The Count service returns how many people in the room

FIGURE 6.8: Count Service

Final results for the test: PASS

6.2 Testing Skill: Smart Banking

Requirements Tested:- Smart button, Log-in Service, balance Service, transaction Service, orders Evens Service,send service,order service,log out service,help service.

1. Power up the button.
2. Next, we run the Jovo skill and Endpoint and ask Alexa about a given service :
   (a) Balance Service
   (b) Transaction Service
   (c) Send Service
   (d) Order Service
   (e) Log-in Service
   (f) Help Service
   (g) Log Out Service

Expected Results:

1. When we invoke the skill name we should be asked to log in.
2. The button should vibrate and we should be able to either short presses or long presses to enter the password.
3. Alexa should allow us to use the services if the password is right and deny if it is wrong.
4. After we log-in we should be able to use the given service :
   (a) Balance Service
   (b) Transaction Service
   (c) Send Service
   (d) Order Service
   (e) Log Out Service
5. when the user order or send the server should generate a random pattern to authenticate the user using the button.

Actual Results:

1. When we invoke the skill name we were able to log in successfully.shown in Figure 6.9.

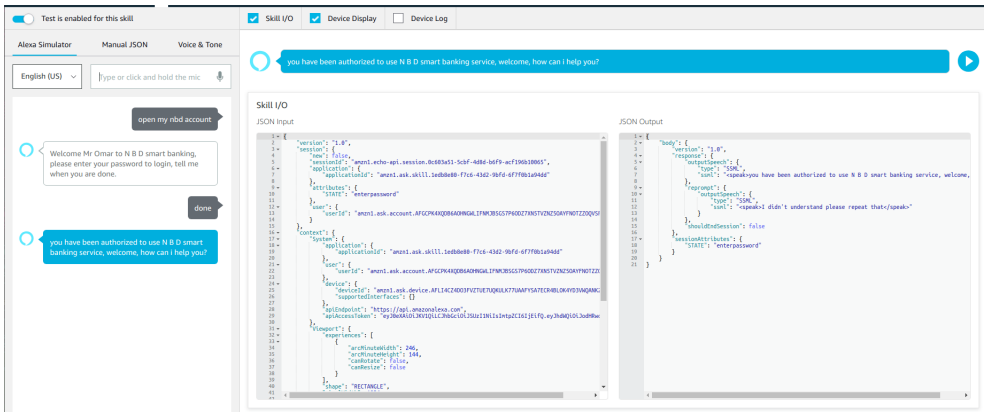


FIGURE 6.9: The Temperature

2. The button vibrates whenever we log in and we were able to enter the password either using short presses or long presses.

3. Alexa allowed us to use the services when the password is right and denied it when it is wrong.

FIGURE 6.10: Firebase auth

4. After we log-in we were able to use the given service :

   (a) Balance Service

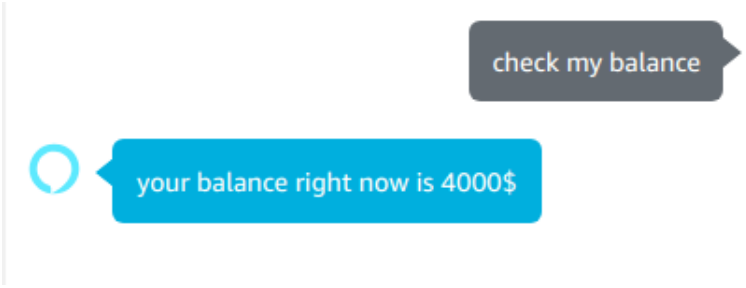


FIGURE 6.11: The Balance

   (b) Transaction Service

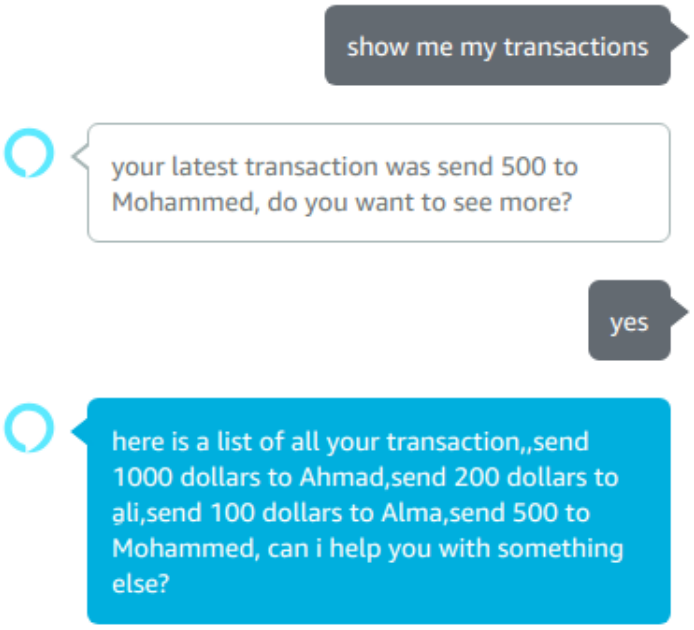


FIGURE 6.12: The Transactions

   (c) Send Service

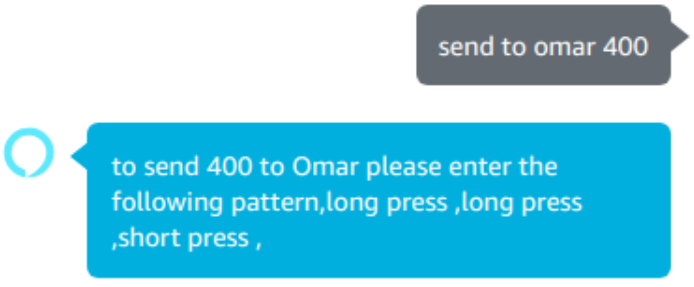


FIGURE 6.13: Send

(d) Order Service

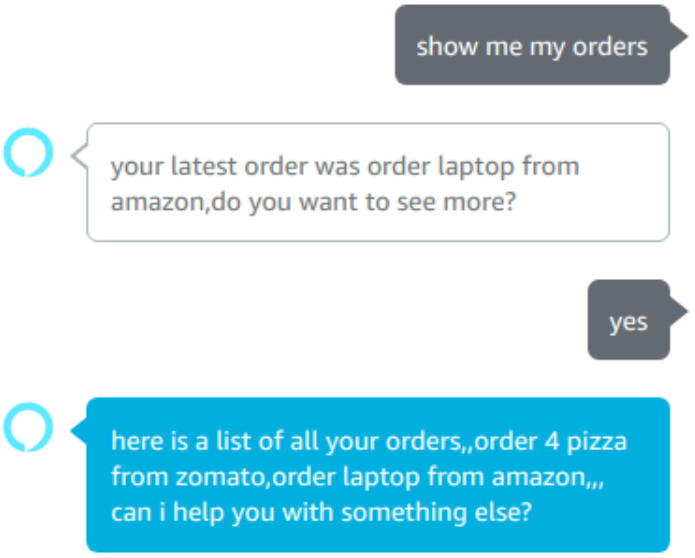


FIGURE 6.14: Order

(e) Orders Service

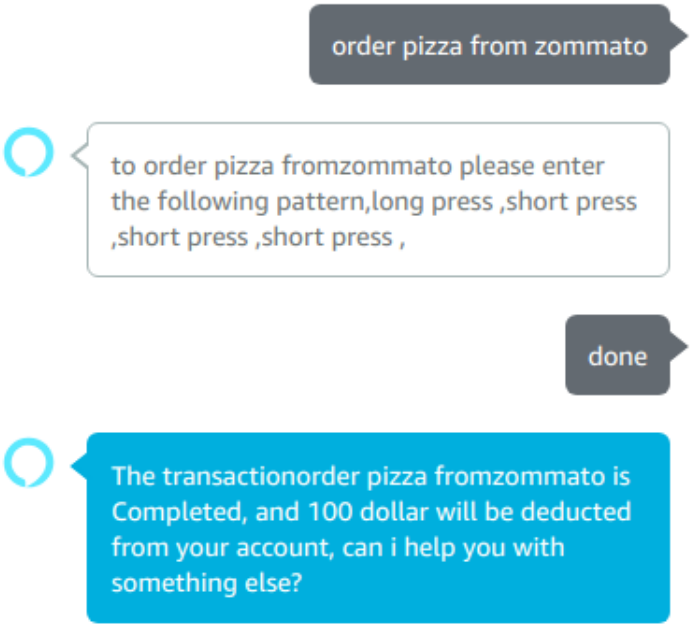


FIGURE 6.15: Orders

(f) Log Out Service

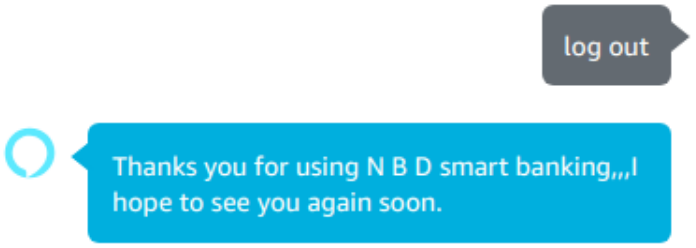


FIGURE 6.16: Log Out

5. when the user order or send the server generated a random pattern to authenticate the user

Final results for the test: PASS

6.3 Testing Skill:Super Sensor

Requirements Tested:- Sensors, Temperature Service, Light Service, Evens Service, Previous Evens Service,Total Water Consummation service.

1. From the Respeaker Sensor code which read all the sensor values.
2. Next, we run the Jovo skill and Endpoint and ask Alexa about a given service :
   (a) Light Service
   (b) Total Water Consummation Service
   (c) Evens Service
   (d) Previous Evens Service
   (e) Temperature Service

Expected Results:

1. Sensor values will be collected and sent to Firebase.
2. Using Google function we will do features extraction in PyAudio
3. Using ML Kit module that we trained we did classification for the collected data.
4. The output should be stored in the Real-time Database thought Firebase

5. When the user invoke the name of the skill when Alexa is running, it should respond
   (a) When the user invoke the Temperature service, it should return the Temperature.
   (b) The Light service should return if the light on or off.
   (c) The Evens service should return the currant event.
   (d) The Previous Evens service should list all of the Previous Evens.
   (e) The Total Water Consummation service should return the Total Water Consummation for the month.

Actual Results:

1. Sensor values has been collected and sent in the Firebase as shown in Figure 6.17.

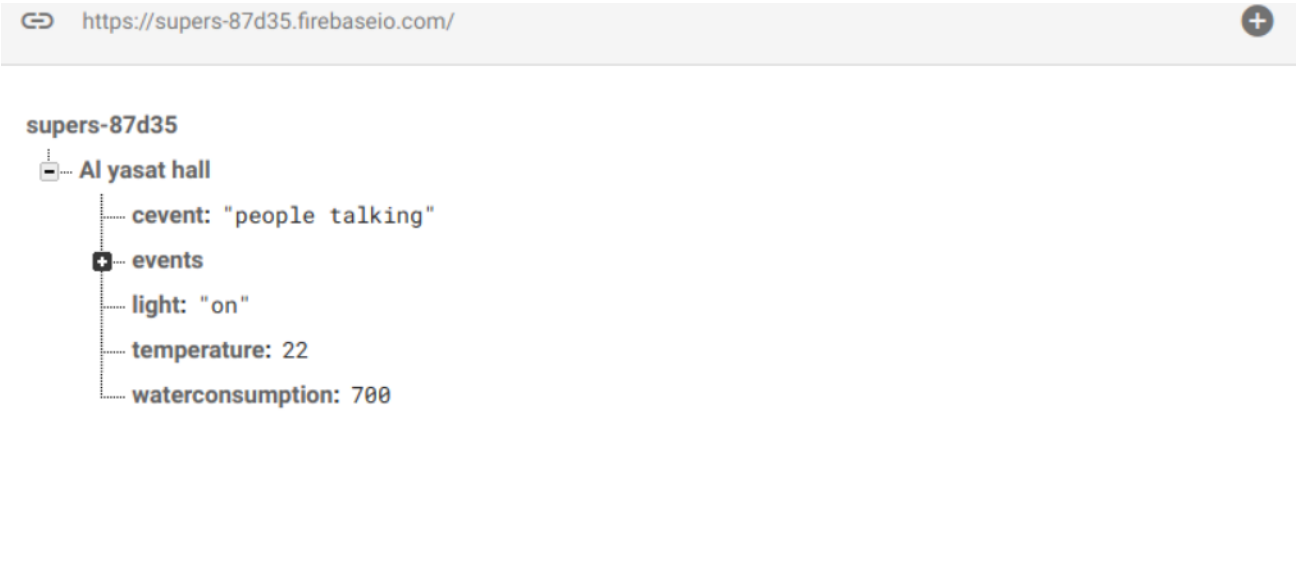


FIGURE 6.17: Sensor Values in Firebase

2. All of the information should be uploaded to the Real-time Database thought Firebase
3. The user invoke the name of the skill while Alexa is running,so it response to the invocation

(a) The Temperature service return the Temperature as shown in Figure 6.18.

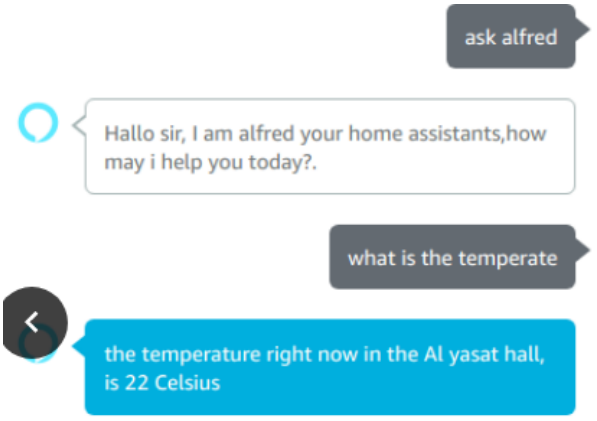


FIGURE 6.18: The Temperature

(b) The Light service return the light status as shown in Figure 6.19.

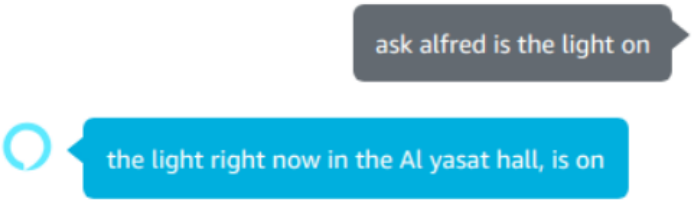


FIGURE 6.19: The Light service

(c) The Evens service return the event that is stored manually as shown in Figure 6.20.

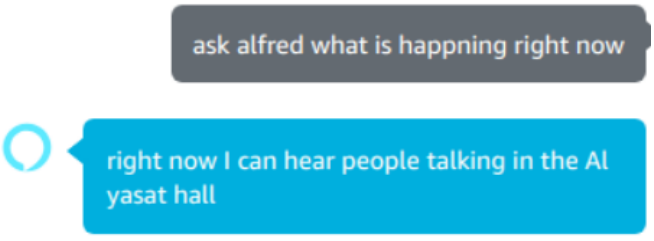


FIGURE 6.20: The Current Events

(d) The Previous Evens service should list all of the Previous Evens that is stored manually as shown in Figure 6.21.

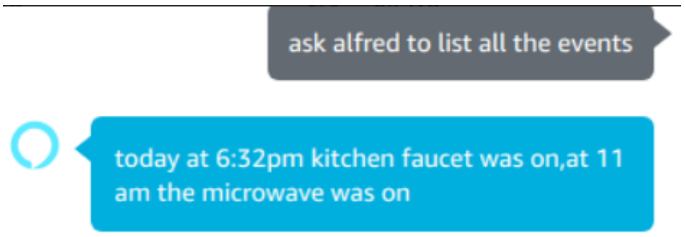


FIGURE 6.21: The Previous Events

(e) The Total Water Consummation service return the Total Water Consummation for the current month as shown in Figure 6.22.

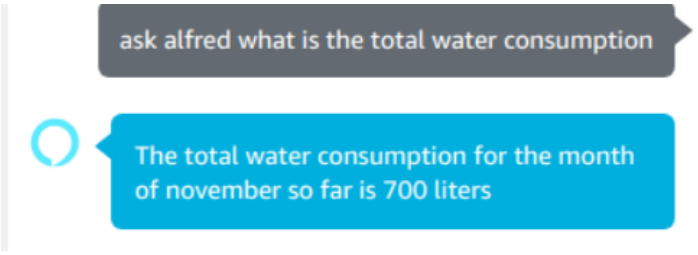


FIGURE 6.22: The Total Water Consummation

Final results for the test: PASS (we did not implement the machine learning part because we did it with the feather M0 implementation)

Chapter 7

Conclusion and Future Work

7.0.1 Summary

The system that we proposed is mainly about helping the Visually Impaired People Of Determination. The project is made by using Voice User Interface, a smart board (Respeaker), smart sensors and deep learning algorithms. We had three main skills, Description of the environment skill, sensing skill and banking skill. We had used the Respeaker board which has 6 Microphone Array along with Amazon Alexa Echo to implement our Voice User Interface. We have also used smart sensors to implement a super sensor which was integrated into our system's circuitry. For the smart banking skill, we manufactured our own smart key chain to double the security measurements by having 2FA. Our skills was made using Angular application, Alexa voice services, Jovo and Firebase. Angular was our interface were we input the static description for the given environment where the system is going to be placed. Alexa voice services alone with Jove were necessary for creating our own Alexa skills as services were provided through the Amazon servers were our skills live, and Jovo was used to hander dialog of every possible conversation. Firebase was used to link it all together as we used its Real-Time Database for our smart key chain system. The testing stage went smoothly and as expected and we have not run into any errors.

7.0.2 Lessons Learned

7.0.2.1 Technical Lessons

In this project we have gained a lot of technical experience when it comes to building a whole system from scratch. The concept of working on such a huge project was new and quite challenging to us, which helped as in gaining a new different prospective on

project making as a whole. The Respeaker board was not easy to operate on, so we did our fair share on debugging to fix everything in order for the board to function properly. That was not the only thing that needed debugging tho, running our project using Alexa voice service was also quite challenging, but we have manage to overcome that. Also, learning about Angular Application and Jovo services was actually amusing and interesting. The overall technical skills that we have gained throughout this project helped us develop our Engineering skills in every aspect possible.

7.0.2.2 Team Work Lessons

Due to being a huge project, we had to work on a strict schedule in order to avoid possible deadline delays and of course bad habits such as procrastination. Our whole team had to really work and do their best at their given tasks in order to to achieve our common goal and to complete our task in the most effective and efficient way. By working in this project we have come to know how hard is it to work on when you have to deliver by the end of the deadline and its a responsibility the falls upon every team member, so its not something that one can do it alone nor is it something that should be taken lightly. the experience that we have gained from working in this project will definitely help us in the future when working on such big projects with groups of different people.

7.0.3 Future Work

After we have finished with our implementation, we thought of things that are missing in our project and its effect on the target user, we also asked our selves in this implementation the most suitable for the Visually Impaired People Of Determination ?
One thing that we have realized is that our main objective in the whole project was to have a smart system that functions as the vision of the Visually Impaired People Of Determination. Our current system has one main problem that is needed to be improved in order to achieve our main goal; the fact that the system isn't mobile. Out system is currently functioning perfectly, but only within a certain environment, so we thought that we need another piece of hardware to do the capturing part of the images; a mobile camera. Having a mobile camera isn't hard, but the problem is we also have to consider having a mobile Voice User Interface too. After some research,we found a hardware piece that can be used as a base to test the idea of transferring our whole project from functioning is static environments only to mobile environments.

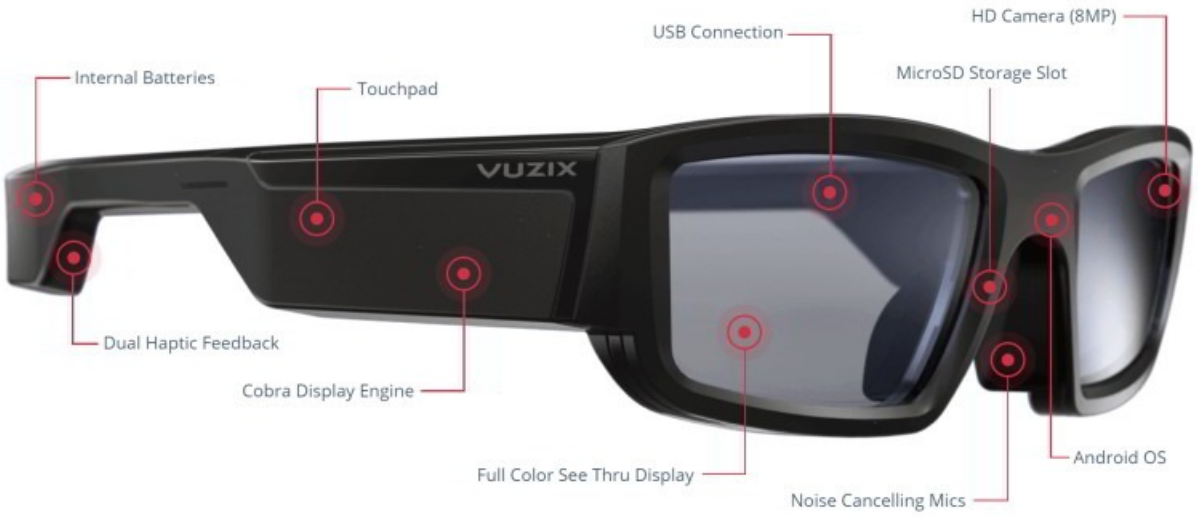


FIGURE 7.1: Smart Glasses

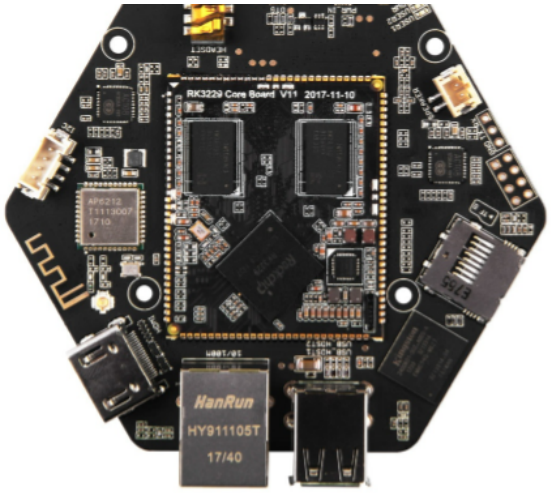
Meet Smart Glasses, a light wearable head glasses that has the following functions :-

- HD Camera
- USB Connection
- Noise Canceling Mics
- Touch Pad
- Internal Batteries
- Android OS

These glasses has almost everything we need to implement our project mobily, we just have to work on some minor changes and that's it.

Appendix A

Respeaker Documentation



Seeed ReSpeaker Core v2 is designed for voice interactive applications. It is based on quad-core ARM Cortex-A7, up to 1.5Ghz, and 1GB RAM on-board. Besides, it features six microphone array with necessary speech algorithm, like DoA(Direction of arrival), BF(Beam-Forming), AEC(Acoustic echo cancellation) and etc.

ReSpeaker Core v2 runs GNU/Linux operation system. It benefits from powerful and active community, we can use lot of existing software/tools for development, testing and deploy, so that rapid product development become available.

ReSpeaker Core v2 is not only designed for makers/enthusiast, but also a turnkey solution for business company. The hardware consists of two parts, one is the minimized SoC module which is small and easy for manufacturing and ready for final product, the other is a bottom board which can be full customizable.

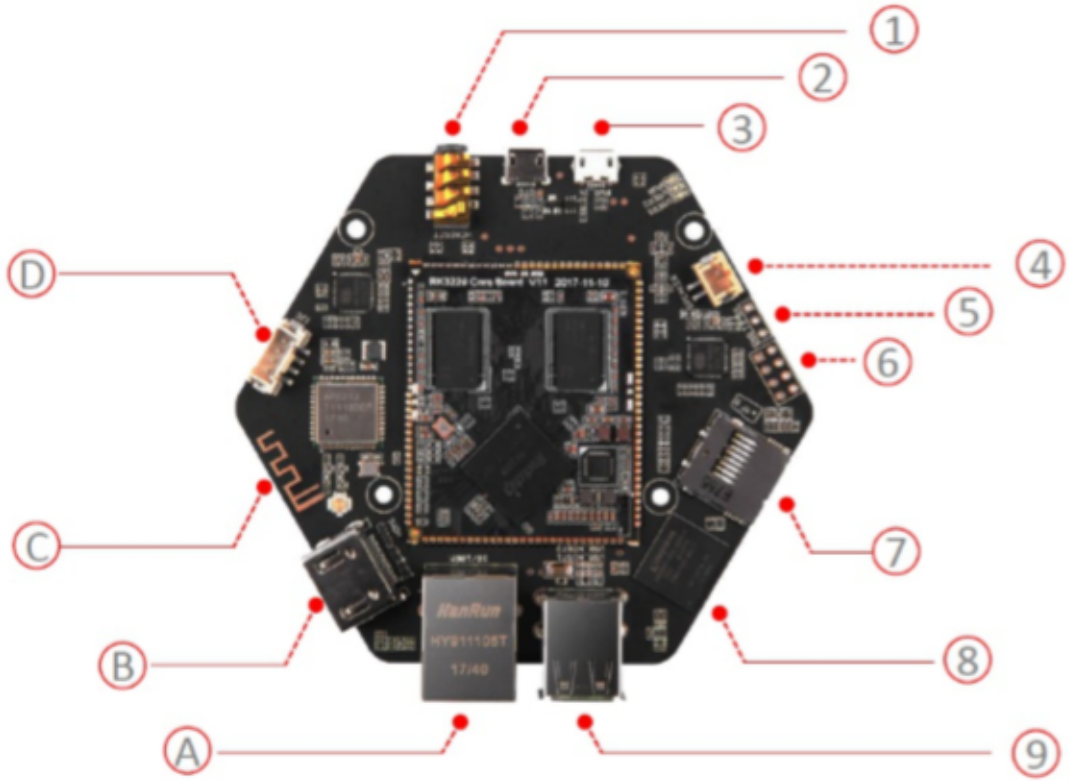
Features

- High performance SoC
- 1GB RAM & 4GB eMMC
- 6 Microphone Array
- USB OTG, USB device
- WiFi b/g/n and BLE 4.0
- Detect range: ~5 meters
- Grove socket for other sensor
- 3.5mm audio jack & JST2.0 connector
- 8 channel ADCs for 6 microphone array and 2 loopback (hardware loopback)

- Debian-based Linux system
- SDK for speech algorithm w ith Full documents
- C++ SDK and Python w rapper
- Speech algorithms and features
- Keyw ord w ake-up
- BF(Beam-Forming)
- DoA (Direction of arrival)
- NS(Noise suppression)
- AEC (Acoustic echo cancellation) and AGC (Automatic gain control)

Specification

Features		
Soc(Rockchip RK3229)	CPU	Quad-Core Cortex-A7,up to 1.5GHz
	GPU	Mali400MP, Support OpenGL ES1.1/2.0
	Memory	1GB RAM(Core Module includes RAM and PMU)
	System	Operating Voltage:3.6-5V
		80 pins on-module
		PMU on-module
Peripheral	Networks	WiFi b/g/n; BLE 4.0; Ethernet
	USB	2 x USB Host; 1 x USB OTG; 1 x USB power
	Grove	1 x Grove socket (I2C and Digital)
	Vedio	HDMI 2.0 with HDCP 1.4/2.2, up to 4K/60Hz
	Audio	6 Microphone Array; 3.5mm Audio Jack; JST2.0 Audio output connector
	Storage	4GB eMMC on-board; SD slot
	Others	12 x RGB LEDs; 8 GPIO pins
Power Consumption	Standby Mode	360mA

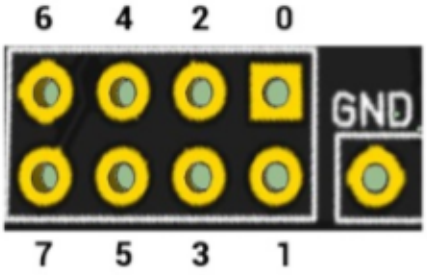
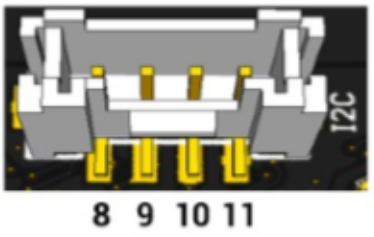


- ① **3.5mm Headphone jack:**
  Output audio. You can plug active speakers or Headphones into this port.
- ② **USB OTG:**
  This USB Port is used to connect to your computer via serial mode of putty (or other serial tools).
- ③ **USB Power Input:**
  This port is used to provide power for Respeaker Core v2.
- ④ **Speaker Jack:**
  Output audio for passive speakers. Jst 2.0 Socket.
- ⑤ **UART:**
  You also can connect the ReSpeaker Core v2 with your computer via this UART port.
- ⑥ **8 Pins GPIO:**
  General Purpose Input Output interface for extended applications.
- ⑦ **SD Card Slot:**
  To plug in micro-SD card.
- ⑧ **eMMC:**
  Embedded Multi Media Card. You can burn the image into eMMC, so that the ReSpeaker Core v2 can boot from the eMMC.
- ⑨ **USB Host:**
  You can plug USB device, such as USB mouse,USB keyboard and USB flash disk into ReSpeaker Core v2 via those two USB hosts.
- Ⓐ **Ethernet:**
  Access to the Internet.

- Ⓑ **HDMI:**
  Output video.
- Ⓒ **Bluetooth and WIFI Antenna:**
  The onboard Antenna is for WIFI and Bluetooth. Also we provide a interface for 2.4G Antenna or PCB Antenna.
- Ⓓ **Grove Socket:**
  Grove Socket for digital or I2C.

Pin Out

Pin index definition for headers :

8 pins header	Grove Socket
	

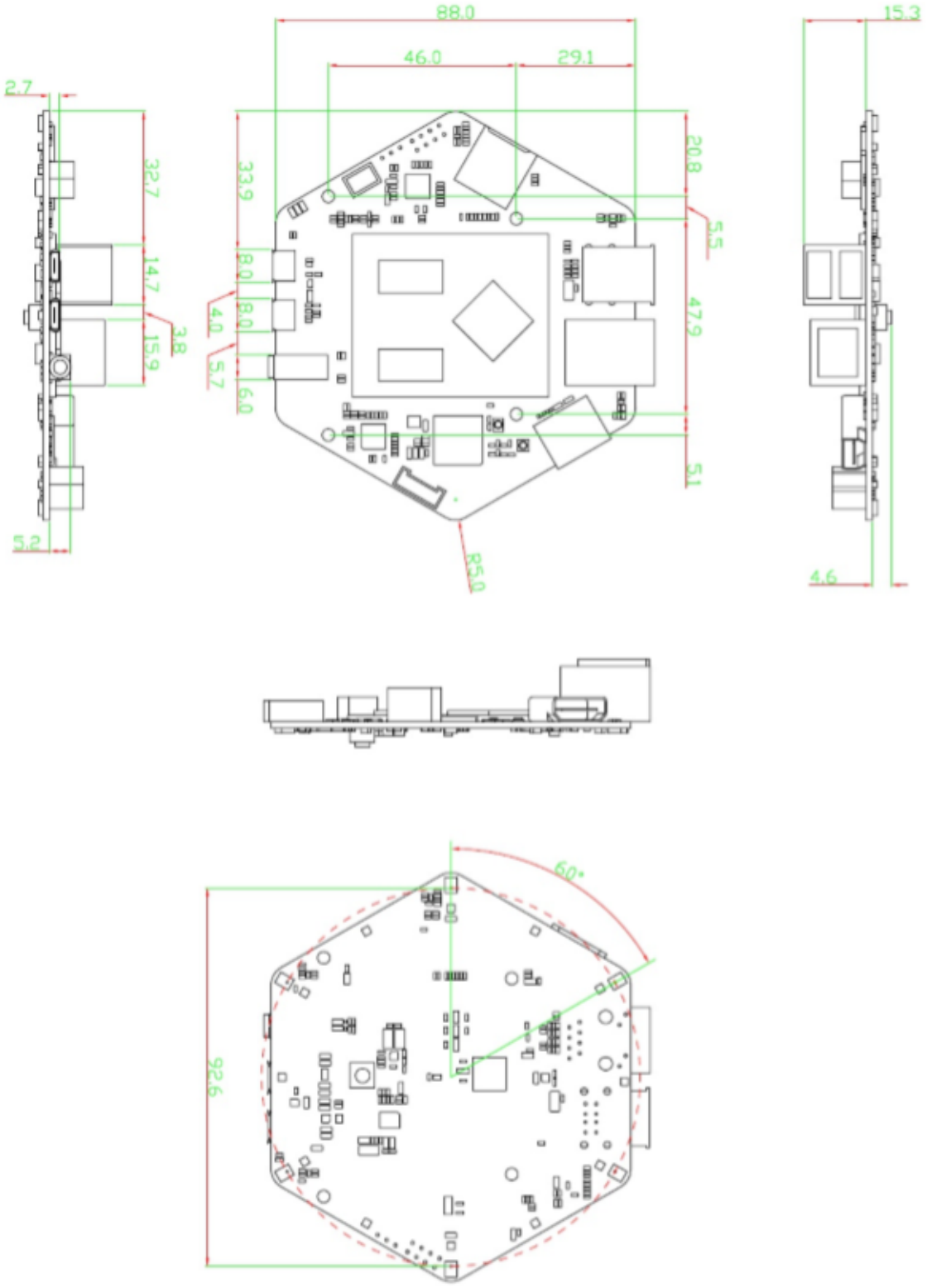
GPIO Pins

MRAA	HEADER PIN INDEX	SYSFS PIN	RK3229 PIN
0	0	1091	GPIO2_D3
1	1	--	VCC
2	2	1043	GPIO1_B3
3	3	1127	GPIO3_D7
4	4	1017	GPIO0_C1
5	5	1067	GPIO2_A3
6	6	--	GND
7	7	1013	GPIO0_B5
8	8	1085	GPIO2_C5
9	9	1084	GPIO2_C4
10	10	--	VCC
11	11	--	GND

I2C Pins

MRAA	HEADER PIN INDEX	SYSFS PIN	RK3229 PIN
0	8	--	I2C2_SCL
0	9	--	I2C2_SDA

Dimensions



Applications

- Smart speaker
- Intelligent voice assistant systems

Appendix B

LSM9DS0 Documentation

Table 19. Accelerometer and gyroscope SAD+Read/Write patterns

Command	SAD[6:1]	SAD[0] = SA0	R/W	SAD+R/W
Read	110101	0	1	11010101 (D5h)
Write	110101	0	0	11010100 (D4h)
Read	110101	1	1	11010111 (D7h)
Write	110101	1	0	11010110 (D6h)

Table 20. Magnetic sensor SAD+Read/Write patterns

Command	SAD[6:2]	SAD[1] = SDO/SA1	SAD[0]	R/W	SAD+R/W
Read	00111	0	0	1	00111001 (39h)
Write	00111	0	0	0	00111000 (38h)
Read	00111	1	0	1	00111101 (3Dh)
Write	00111	1	0	0	00111100 (3Ch)

Table 21. Accelerometer and gyroscope register address map

Name	Type	Register address		Default	Note
		Hex	Binary		
Reserved	--	00-03	--	--	Reserved
ACT_THS	r/w	04	00000100	00000000	
ACT_DUR	r/w	05	00000101	00000000	
INT_GEN_CFG_XL	r/w	06	00000110	00000000	
INT_GEN_THS_X_XL	r/w	07	00000111	00000000	
INT_GEN_THS_Y_XL	r/w	08	00001000	00000000	
INT_GEN_THS_Z_XL	r/w	09	00001001	00000000	
INT_GEN_DUR_XL	r/w	0A	00001010	00000000	
REFERENCE_G	r/w	0B	00001011	00000000	
INT1_CTRL	r/w	0C	00001100	00000000	
INT2_CTRL	r/w	0D	00001101	00000000	
Reserved	--	0E	--	--	Reserved
WHO_AM_I	r	0F	00001111	01101000	
CTRL_REG1_G	r/w	10	00010000	00000000	
CTRL_REG2_G	r/w	11	00010001	00000000	
CTRL_REG3_G	r/w	12	00010010	00000000	
ORIENT_CFG_G	r/w	13	00010011	00000000	
INT_GEN_SRC_G	r	14	00010100	output	
OUT_TEMP_L	r	15	00010101	output	
OUT_TEMP_H	r	16	00010110	output	
STATUS_REG	r	17	00010111	output	
OUT_X_L_G	r	18	00011000	output	
OUT_X_H_G	r	19	00011001	output	
OUT_Y_L_G	r	1A	00011010	output	
OUT_Y_H_G	r	1B	00011011	output	
OUT_Z_L_G	r	1C	00011100	output	
OUT_Z_H_G	r	1D	00011101	output	
CTRL_REG4	r/w	1E	00011110	00111000	
CTRL_REG5_XL	r/w	1F	00011111	00111000	

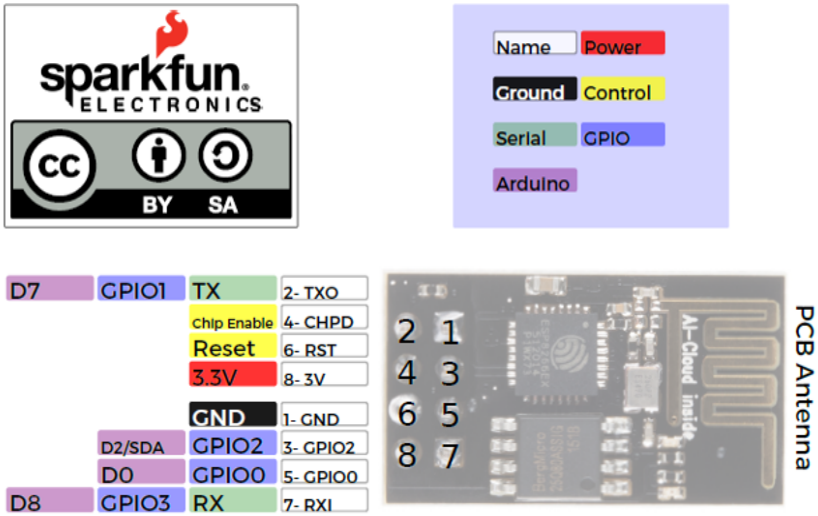
Table 17. Register address map

Name	Slave Address	Type	Register address		Default
			Hex	Binary	
Reserved	Table 16	--	00-0E	--	--
WHO_AM_I_G	Table 16	r	0F	000 1111	11010100
Reserved	Table 16	--	10-1F	--	--
CTRL_REG1_G	Table 16	rw	20	010 0000	00000111
CTRL_REG2_G	Table 16	rw	21	010 0001	00000000
CTRL_REG3_G	Table 16	rw	22	010 0010	00000000
CTRL_REG4_G	Table 16	rw	23	010 0011	00000000
CTRL_REG5_G	Table 16	rw	24	010 0100	00000000
REFERENCE_G	Table 16	rw	25	010 0101	00000000
Reserved	Table 16	--	26	--	--
STATUS_REG_G	Table 16	r	27	010 0111	output
OUT_X_L_G	Table 16	r	28	010 1000	output
OUT_X_H_G	Table 16	r	29	010 1001	output
OUT_Y_L_G	Table 16	r	2A	010 1010	output
OUT_Y_H_G	Table 16	r	2B	010 1011	output
OUT_Z_L_G	Table 16	r	2C	010 1100	output
OUT_Z_H_G	Table 16	r	2D	010 1101	output
FIFO_CTRL_REG_G	Table 16	rw	2E	010 1110	00000000
FIFO_SRC_REG_G	Table 16	r	2F	010 1111	output
INT1_CFG_G	Table 16	rw	30	011 0000	00000000
INT1_SRC_G	Table 16	r	31	011 0001	output
INT1_TSH_XH_G	Table 16	rw	32	011 0010	00000000
INT1_TSH_XL_G	Table 16	rw	33	011 0011	00000000
INT1_TSH_YH_G	Table 16	rw	34	011 0100	00000000
INT1_TSH_YL_G	Table 16	rw	35	011 0101	00000000
INT1_TSH_ZH_G	Table 16	rw	36	011 0110	00000000
INT1_TSH_ZL_G	Table 16	rw	37	011 0111	00000000
INT1_DURATION_G	Table 16	rw	38	011 1000	00000000
Reserved	Table 15	--	00-04	--	--

Appendix C

ESP8266 Documentation

ESP8266 Module (WRL-13678)



Power
VCC-3.0-3.6V
Standby ~ 0.9uA
Running ~60-215mA,
Average ~ 80mA

Wifi Features
802.11 b/g/n
2.4GHz
WPA/WPA2
Wifi Direct

+20dBm output power (802.11b)

I/O Features
Integrated TCP/IP
Integrated TR switch, LNA, balun

Memory/Speed Features
80MHz
64KB Instruction RAM
96KB data RAM
64K boot ROM
1MB* Flash Memory

Basic Connection
VCC - 3.3V
GND - GND
TX - RX on Arduino or FTDI
RX - TX on ARduino or FTDI
Chip Enable - 3.3V

Default Baud Rate
11520* 8N1

LEDs
Red: Power
Blue: TX

*milage may vary on different version of the board

AT Command Usage

Commands are case sensitive and should end with /r/n

Commands may use 1 or more of these types
Set = AT+<x>=<...> - Sets the value
Inquiry = AT+<x>? - See what the value is set at
Test = AT+<x>=? - See the possible options
Execute = AT+<x> - Execute a command

Commands with * have been depreciated in favor of COMMAND_CUR and COMMAND_DEF. CUR will not write the value to flash. DEF will write the valueto flash and be usedas the default in the future.

AT Command List

AT - Attention
AT+RST - Reset the board
AT+GMR - Firmware version
AT+CWMODE* - Operating Mode
1. Client
2. Access Point
3. Client and Access Point
AT+CWJAP*=<ssid>,<pwd> - Join network
AT+CWLAP - View available networks
AT+CWQAP - Disconnect from network
AT+CWSAP*=<ssid>,<pwd><chl><ecn> - Set up access point
0. Open. No security
1. WEP
2. WPA_PSK
3. WPA2_PSK
4. WPA_WPA2_PSK
AT+CWLIF - Show assigned IP addresses as access point
AT+CIPSTATUS - Show current status as socket client or server
AT+CIPSTART=<type>,<addr>,<port> - Connect to socket server
IP is fixed at 192.168.4.1, mask is fixed at 255.255.255.0
If CIPMUX is set to multichanel add <id> to beginning of string
AT+CIPCLOSE - Close socket connection
AT+CIFSR - Show assigned IP address when connected to network
AT+CIPMUX=<mode> - Set connection
0. Single Connection
1. Multi-Channel Connection
AT+CIPSERVER=<mode>[,<port>](AT+CIPMUX=1) - Default port is 333
0. Close the Socket Server
1. Open the Socket Server
AT+CIPMODE=<mode> - Set transpartent mode
Data received will be sent to serial port as
0. +IPD,<connection channel>,<length>format (AT+CIPMUX=[0,1])
1. Data stream (AT+CIPMUX=0)
AT+CIPSTO=<time> - Set auto socket client disconnect timout from 1-28800s

Example commands
AT+CWMODE=? //View options for mode (test)
AT+CWMODE=3 //Set mode to client and access modes (set)
AT+CWLAP //View available networks (execute)
AT+CWJAP = "ssid","password" //Join network (set)
AT+CWJAP? //View the current network (inquiry)
AT+CIFSR //Show IP address (execute)
AT+CWQAP //Disconnect from network (execute)
AT+CWSAP="apoint","pass",11,0//Setup an open access point (set)
AT+CWLIF //Show devices connected to access point

Bibliography

[1] Theodoros Giannakopoulos. pyaudioanalysis: An open-source python library for audio signal analysis. *PLOS ONE*, 10:e0144610, 12 2015. doi: 10.1371/journal.pone.0144610.

[2] Seeed Studio. Respeaker core v2.0. 2017. URL http://wiki.seeedstudio.com/ReSpeaker_Core_v2.0/.

[3] Jovo. Build an alexa skill in node.js with jovo. 2018. URL https://www.jovo.tech/tutorials/alexa-skill-tutorial-nodejs.

[4] World Health Organization. Blindness and vision impairment. 2018. URL (http://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment).

[5] gulf news. 115m blind people by 2050. 2017. URL (https://gulfnews.com/lifestyle/health-fitness/115m-blind-people-by-2050-1.2068245).

[6] Albayan News. achievements of people of determination is a prove that human will has no limits. 2017. URL (https://www.albayan.ae/across-the-uae/news-and-reports/2017-04-19-1.2920526).

[7] Margaret Rouse. Alexa voice services (avs). URL https://whatis.techtarget.com/definition/Alexa-Voice-Services-AVS.

[8] Amazon. Alexa voice service. URL https://developer.amazon.com/alexa-voice-service.

[9] Jaycon Systems. Getting started with the esp8266 esp-01. URL https://www.instructables.com/id/Getting-Started-With-the-ESP8266-ESP-01/.

[10] Github. github. 2017. URL (https://github.com//).

[11] fritzing. fritzing. URL (http://fritzing.org/home/).

[12] Github. Mraa libarary. 2017. URL https://github.com/intel-iot-devkit/mraa.

[13] UPM. Upm sensor framework for iot depelpvers. 2017. URL https://upm.mraa.io/Documentation/docindex.html.

[14] FireBase Team. Getting started with firebase. . URL https://firebase.google.com/products/realtime-database/.

[15] James Tamplin. Firebase expands to become a unified app platform. 2016. URL https://firebase.googleblog.com/2016/05/firebase-expands-to-become-unified-app-platform.html.

[16] FireBase Team. Firestore. . URL https://firebase.google.com/products/firestore/.

[17] FireBase Team. Firebase functions. . URL https://firebase.google.com/products/functions/.

[18] FireBase Team. Firebase authentication. . URL https://firebase.google.com/products/auth/.

[19] FireBase Team. Firebase hosting. . URL https://firebase.google.com/products/hosting/.

[20] FireBase Team. Firebase storage. . URL https://firebase.google.com/products/storage/.

[21] FireBase Team. Firebase ml kit. . URL https://firebase.google.com/products/ml-kit/.

[22] Angular.JS Team. Getting started with angular.js. . URL https://angularjs.org/.

[23] Angular Team. Getting started with angular. . URL https://blog.angular.io/.

[24] Angular Team. Angular architecture. . URL https://www.simplilearn.com/angularjs-vs-angular-2-vs-angular-4-differences-article.

[25] Angular.JS Team. Angular.js architecture. . URL https://medium.freecodecamp.org/why-is-angularjs-the-most-preferred-framework-for-software-devel

[26] Angular.JS Team. Angular.js architecture. 2014. URL https://techterms.com/definition/javascript.

[27] Angular Team. Angular architecture. . URL https://www.typescriptlang.org/docs/home.html.

[28] Angular Team. Angular lifecycle hooks. . URL https://angular.io/guide/lifecycle-hooks.

[29] Amal Shajan. Smart glass for the blind. 2018. URL (https://create.arduino.cc/projecthub/B45i/talking-smart-glass-for-the-blind-87d31e?ref=tag&ref_id=blind&offset=2).

[30] Team MeRo. Blind stick navigator. 2016. URL (https://create.arduino.cc/projecthub/mero/blind-stick-navigator-b119f5?ref=tag&ref_id=blind&offset=0).

[31] Team Be My Eyes. 2015. URL (https://www.bemyeyes.com/).

[32] Emirates News Agency. Hh sheikh hamdan saying. 2017. URL http://wam.ae/ar/details/1395302655053.

[33] Github. pyaudioanalysis. 2017. URL (https://github.com/tyiannak/pyAudioAnalysis/).

[34] Adafruit Industries. Adafruit feather m0 basic proto - atsamd21 cortex m0. 2017. URL https://www.adafruit.com/product/2772.

[35] Adafruit Industries. Adafruit lsm9ds0. 2017. URL https://learn.adafruit.com/adafruit-lsm9ds1-accelerometer-plus-gyro-plus-magnetometer-9-dof-breakout/overview.

[36] Adafruit Industries. Adafruit sph0645lm4h. 2017. URL https://www.adafruit.com/product/3421.

[37] Fangcloud. Respeaker image cloud. 2018. URL https://v2.fangcloud.com/share/7395fd138a1cab496fd4792fe5?folder_id=188000207914&lang=en.

[38] erikhopf amzn. Instruction to create security profile. 2017. URL https://github.com/alexa/alexa-avs-sample-app/wiki/Create-Security-Profile.

[39] Adafruit Industries. Adafruit bme280 humidity and barometric pressure and temperature sensor breakout. 2017. URL https://www.marutsu.co.jp/contents/shop/marutsu/ds/adafruit-bme280-humidity-barometric-pressure-temperature-sensor-breakout.pdf.

[40] Adafruit Industries. Lux sensor - tsl2561 light sensor. 2017. URL https://www.adafruit.com/product/439.

[41] Github. Use mraa and upm on respeaker v2. 2017. URL https://github.com/respeaker/get_started_with_respeaker/blob/master/docs/ReSpeaker_Core_V2/mraa_upm.md.

[42] Omar Al Tawil. Python library for lsm9ds1 i2c. 2018. URL https://github.com/omar7altawil/LSM9DS1-I2C.