

**BANGLA HOME ASSISTANT USING CMU SPHINX OFFLINE  
SPEECH RECOGNITION**

By

**Sourav Das**  
**ID: 161-15-7243**  
AND

**Amit Kundu**  
**ID: 161-15-7149**  
AND

**Al-Shahriar**  
**ID: 161-15-7188**

This Report Presented in Partial Fulfillment of the Requirements for the  
Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

**Prof. Dr. Syed Akhter Hossain**  
**Head**  
Department of CSE  
Daffodil International University



**DAFFODIL INTERNATIONAL UNIVERSITY**

**DHAKA, BANGLADESH**

**DECEMBER 2019**

## **APPROVAL**

This Project titled “**Bangla Home Assistant Using CMU Sphinx Offline Speech Recognition**” , submitted by Sourav Das, ID No: 161-15-7243, Amit Kundu, ID No: 161-15-7149 and Al-Shahriar, ID No: 161-15-7188 to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 07 December 2019.

## **BOARD OF EXAMINERS**




---

**Dr. Syed Akhter Hossain**

**Professor and Head**

Department of Computer Science and Engineering  
Faculty of Science & Information Technology  
Daffodil International University

**Chairman**



---

**Abdus Sattar**

**Assistant Professor**

Department of Computer Science and Engineering  
Faculty of Science & Information Technology  
Daffodil International University

**Internal Examiner**



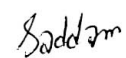
---

**Shaon Bhatta Shuvo**

**Senior Lecturer**

Department of Computer Science and Engineering  
Faculty of Science & Information Technology  
Daffodil International University

**Internal Examiner**



---

**Dr. Md. Saddam Hossain**

**Assistant Professor**

Department of Computer Science and Engineering  
United International University

**External Examiner**

## DECLARATION

We hereby declare that this project has been done by us under the supervision of **Prof. Dr Syed Akhter Hossain, Head, Department of CSE** Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for the award of any degree or diploma.

**Supervised to:**



---

**Prof. Dr Syed Akhter Hossain**  
Head  
Department of CSE  
Daffodil International University

**Submitted by:**



---

**Sourav Das**  
**ID: 161-15-7243**  
Department of CSE  
Daffodil International University



---

**Amit Kundu**  
**ID: 161-15-7149**  
Department of CSE  
Daffodil International University



---

**Al-Shahriar**  
**ID: 161-15-7188**  
Department of CSE  
Daffodil International University

## ACKNOWLEDGEMENT

First, we express our heartiest thanks and gratitude to Almighty God for His divine blessing makes us possible to complete the final year project/internship successfully.

We are grateful and wish our profound our indebtedness to Prof. Dr Syed Akhter Hossain, Head, Department of CSE Daffodil International University, Dhaka. Deep Knowledge & keen interest of our supervisor in the field of “*NLP & IOT*” to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

We would like to express our heartiest gratitude to Mr. Fahad Faisal, Shams Sunny, and Head, Department of CSE, for his kind help to finish our project and also to other faculty members and the staff of CSE department of Daffodil International University.

We would like to thank our entire course mates at Daffodil International University, who took part in this discussion while completing the course work.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

## **ABSTRACT**

We live in an era of science and technology. Our everyday life is influenced in various ways by technologies. Anywhere we look, we see the use of technology. With the increase of modern technologies, we are going to a point where almost every work is done by our tech devices. And the way of interactions has also been changed. Now various types of interactions are being developed which makes using our devices easier than ever. Such a type of interaction is speech recognitions. It is now very popular in the current era because all the tech giants are focusing on speech recognition with their “Personal assistants” like Google Assistant, Amazon Alexa, and Microsoft Cortana. In a speech recognition system, the user uses his/her voice to give command to a device. But sadly, all of these works are being done in English or other popular languages. Our mother tongue, Bengali was left out. So, our goal was to make a voice interactable device that can be used offline for executing tasks. And all of these would be in Bengali. So, we did some research on speech recognitions and made a recognizer that understands commands given in Bengali. And with that, we made a home assistant device so that we can bring out the maximum usability of the recognizer. Because when you are at home, doing some other work, the voice recognizer would be very much helpful. In the future, incorporation of AI with machine learning will extend the idea further.

# TABLE OF CONTENT

<b>CONTENTS</b>	<b>Page No</b>
Approval & Board of Examiners	i
Declaration	ii
Acknowledgements	iii
Abstract	iv
Table of Contents	v-vi
List of Figures	vii
List of Tables	viii
<b>CHAPTER:</b>	
<b>CHAPTER 1: INTRODUCTION</b>	<b>1-3</b>
1.1 Introduction	1
1.2 Motivation	1
1.3 Objectives	2
1.4 Expected Outcome	2
1.5 Report Layout	2-3
<b>CHAPTER 2: BACKGROUND</b>	<b>4-5</b>
2.1 Introduction	4
2.2 Related Works	4
2.3 Comparative Studies	4-5
2.4 Scope of the Problem	5
2.5 Challenges	5
<b>CHAPTER 3: REQUIREMENTS SPECIFICATION</b>	<b>6-17</b>
3.1 Business Process Modeling	6-7

3.2 Requirement Collection and Analysis	8-9
3.3 Use Case Modeling and Description	9-13
3.4 Logical Data Model	14-16
3.5 Design Requirements	17
<b>CHAPTER 4: DESIGN SPECIFICATION</b>	<b>18-31</b>
4.1 Front-end Design	18-20
4.2 Back-end Design	20-30
4.3 Interaction Design and UX	30
4.4 Implementation Requirements	30-31
<b>CHAPTER 5: IMPLEMENTATION AND TESTING</b>	<b>32-35</b>
5.1 Implementation of Database	32
5.2 Implementation of Front-end Design	32-33
5.3 Implementation of Interactions	34
5.4 Testing Implementation	34-35
5.5 Test Results and Reports	35
<b>CHAPTER 6: CONCLUSION AND FUTURE SCOPE</b>	<b>36</b>
6.1 Discussion and Conclusion	36
6.2 Scope for Further Developments	36
<b>REFERENCES</b>	<b>38-39</b>
<b>APPENDIX</b>	<b>37</b>
Appendix A: Project Reflection	37
Appendix B: Related Diagrams	37

## LIST OF FIGURES

<b>FIGURES</b>	<b>PAGE NO</b>
Figure 3.1.1: BPM of Bangla Voice assistance app	7
Figure 3.3.1: Use case diagram of Voice Assistant APP	10
Figure 3.4.1: Full Flowchart of the system	14
Figure 3.4.2: Flowchart recognizer	15
Figure 3.4.3: Acoustic model training diagram	16
Figure 4.1.1: Stating page design in Android Studio	18
Figure 4.1.2: All list of commands activity design	19
Figure 4.1.3: User prompt to save command	20
Figure 4.2.1.1: Voice to phoneset conversation [4]	21
Figure 4.2.1.2: Required format for text2wfreq to work [8]	22
Figure 4.2.1.3: Final Acoustic Model files.	24
Figure: 4.2.3.1 Arduino connection with Bluetooth and Relay module	29
Figure 5.1.1: Implementation of database	32
Figure 5.2.1: Front-end design	32
Figure 5.2.2: Saving Command popup window	33
Figure 5.2.3: Command list	33



## **LIST OF TABLES**

<b>TABLES</b>	<b>PAGE NO</b>
Table 1: Use Case Description of Record Voice	10
Table 2: Use Case Description of Control Bluetooth	11
Table 3: Use Case Description of Recognize voice	11
Table 4: Use Case Description of Save Voice	12
Table 5: Use Case Description of Send in Microcontroller	12
Table 6: Use Case Description of Recognize voice	13
Table 7: List of language model	22
Table 8: List of Phonetic Dictionary	23
Table 9: Result summary	35

# CHAPTER 1

## INTRODUCTION

### 1.1 Introduction

The smart home lets you control a range of connected devices in your home from a smartphone [1]. Nowadays home automation system has acquired more popularity and it changes the living way of life. By using a smartphone anyone can control their smart home easily. Saying a simple voice command, it will perform various obligations of any home appliances. Before that, we have to do all that by ourselves. But for bigger houses, it is quite hard. And currently available home assistance devices use English for voice commands. We aim to change this.

### 1.2 Motivation

Bengali is one of the used and richest languages in the world. 98% of Bangladeshis are fluent in Bengali [2]. Though it sounds shameful, most of our people cannot speak English fluently. So, we wanted to implement speech recognition in Bangla. Although it is not new but there was very little implementation of it. So, we found the best possible implementation which will be available to the mass. We used the recognizer into an Android app and pair with a microcontroller to make a smart home assistant that recognizes Bangla. The revolution of IoT has begun and people are finding countless ways to integrate embedded systems into their products. Homes aren't any different. Home automation isn't something new, but It is hardly seen in developing countries like Bangladesh. And we are trying to achieve that same with much fewer difficulties and cost-efficiently. Also, we are trying to make it easier for everyone to use.

### 1.3 Objective

Our main goal was to implement the recognizer into an app and make a smart home device by achieving these milestones:

- To create a speech recognizer that recognizes the Bengali Language.

- To able to detect discrete commands given in Bengali.
- To able to record/delete and modify command
- To able to execute actions via an embedded device

## **1.4 Expected Outcome**

By creating a speech recognizer new horizon will be opened for speech recognition with Bengali. We create an app that can control all our home appliances like turning on and off lights, fans and various electronics just by saying a command. Our app will connect to a microcontroller device which will be connected by Bluetooth connection. Anyone can control their home appliance by our app using the Bengali command. The app user can control command by recording, deleting and modifying the recognizer command.

## **1.5 Report Layout**

### **Chapter 1: Introduction**

This chapter contains an introduction, Motivation, Objective, Expected Outcome and finally the Report Layout. The reader will get a clear scenario of our project after reading this chapter.

### **Chapter 2: Background**

In this chapter, we discuss the background context of this project. We also discuss the related work, comparative studies, scope of problems. Finally, we discuss the challenges we face to complete this project.

### **Chapter3: Requirement Specification**

In terms of importance, this is a second important chapter in this report. In this chapter we try to present the project context more logically, for example, with diagram, flow charts and other visual methods. In this chapter, we discuss BPM, Requirement collection and analysis, the use case modelling, use case description, and logical data model. Finally, we discuss the design requirements of this project.

#### **Chapter 4: Design Specification**

In this chapter, we focus on the Front-end design and Back-end design. In the back-end design, we discuss how CMU Sphinx work, the App configuration and Microcontroller Configuration. Finally, we discuss the Interaction Design & UX and Implementations Requirements.

#### **Chapter 5: Implementation and Testing**

In this chapter, we discuss the implementation of the database, Implementation of Front-end design, Interaction and testing. In the end, there was a test result and Report.

#### **Chapter 6: Conclusion and Future Scope**

This chapter is all about the conclusion and the scope for further developments of this project.

## **CHAPTER 2**

### **Background**

#### **2.1 Introduction**

Works in Speech Recognition, Home automation, Microcontrollers aren't something new. Many people have worked in these fields. We have reviewed various works on Bengali speech recognition and some works on home automation. But there was no work on the implementation of speech recognition. Various projects were made when people used a recognizer to demonstrate the output. But we found none with actual real-life usage. So, we wanted to take those works, improve them and use it in a real-life scenario.

#### **2.2 Related Works**

There are many works related to our project. But there isn't any exact similar project. We researched a lot of projects and found similar works that are a part of our project. For example, building the speech Recognizer part, building the android app and executing everything with an Arduino.

There's a work on CMU sphinx [3] which is the Speech Recognition framework we used for building our recognizer named Real-Time Bengali Speech to Text

Conversion using CMU Sphinx [4]. They also used CMUSphinx to demonstrate the implementation of Bengali speech recognition with Java with sphinx4 [5].

There are two other works that we followed for home automation. They are WEB APPLICATION BASED WIRELESS HOME AUTOMATION SYSTEM [6] and Home Automation via Bluetooth using Android Application [7].

#### **2.3 Comparative Studies**

Our project mainly works in three parts. The recognizer, the app and the Arduino (Microcontroller). There are lots of works mentioned in the previous chapter. The

works in Bangla Speech Recognition were done by some people in the past but they just demonstrate the technique. There was no real-life usage of those projects. We improved and optimized their work and put that into a real-life scenario. Then our app is totally custom designed. So, it is one of its kind. And all the usage of microcontrollers is just for executing the actions. In short, this system is the first of its kind to our knowledge.

## **2.4 Scope of the Problem**

After completing the project, a new horizon will open for speech recognition with the Bengali language. People can contribute to the speech recognizer to improve the accuracy and detection of the Bengali command. Another scope this smart home assistance will be cheaper than other smart assistance and it will compete with the assistance of the others

## **2.5 Challenges**

- The first challenge for our speech recognizer will be its accuracy improvement.
- The accuracy of the speech recognizer must be above 95% accuracy.
- For improving accuracy, we will create a bigger collection of voice data.
- About 5+ hours of recording voice data collection needed for the accuracy improvement

## **CHAPTER 3**

### **Requirement Specification**

#### **3.1 Business Process Modeling**

Business Process Modeling or BMP is a process of constructing a structural view of a system or process. It includes some process, starts, ends and some symbol, condition as like a flow chart [12]. In this process we can see that at first, the user has to record voice, then the voice is passed to the CMUSphinx. After then CMUSphinx generates an output and pass it to the system. The app checks the current state of the application. If the saving mood is on the command is saved to the database and if the Bluetooth is enabled, it will try to find a match from the database. If a match found it will send the command (ON/OFF) and the equivalent pin no to the Arduino. Then the Arduino will execute the command. BPM of Bangla Voice assistance app is shown in figure 3.1.1

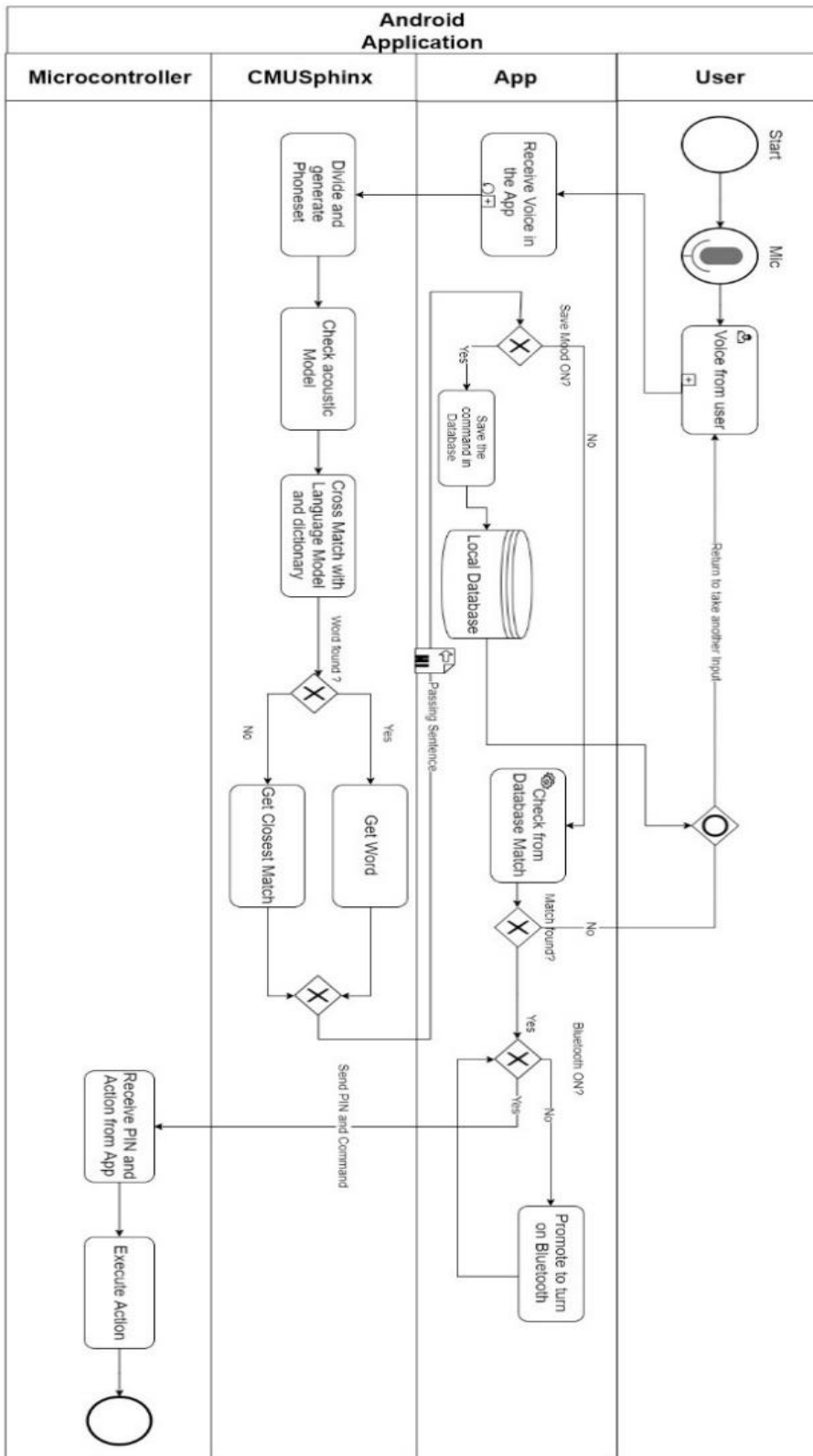


Figure 3.1.1: BPM of Bangla Voice assistance app



In this figure the whole process of the system is described. From the beginning where the user gave the command to the stage where the command was executed is shown in the figure. The four stages indicates the four key component of this system.

First the user will gave command to the android app. The app will use the CMU Sphinx Engine to process the given audio and generate an output. The output is processed further processed and a command is given to the microcontroller for command excitation.

### **3.2 Requirement Collection and Analysis**

This project has three main parts. So, requirement collection and analysis will be different for each part. Basically, they are the Recognizer, an Android Application, and A Microcontroller.

Everything was done in 2 Systems. The specifications are the following:

System 1:

- OS: Arch Linux
- Processor: i5-7500
- RAM: 32GB
- GPU: Nvidia GTX 1070Ti
- SSD: 240GB

System 2:

- OS: Windows 10
- Processor: Intel i3-4130
- RAM: 8GB
- SSD: 250GB

#### ***Recognizer***

To build the recognizer, we needed:

- Text Corps: Raw text data (Conversations, Articles, Commands).
- Voice samples and Data: For training acoustic model.

Tools & Programming Language we needed:

- Python 3.7: To convert text into the required format.
- Python 2.6: To run MMIE training scripts.
- Audacity 2.1.1
- Praat 6.1.03
- TensorFlow
  - **Tesnsor2tensor**: To run g2p-seq2seq [9]
- Bash: To auto-setup CMU Sphinx.
- CMU Sphinx Framework
  - **text2wfreq** for converting raw data into vocabulary file.
  - **text2idngram** for generating an ARPA format language model.
  - **g2p-seq2seq** for generating initial Dictionary.
  - **Sphinxtrain** [10] for training the acoustic model.
  - **Sphinxtrain** [11] for training the acoustic model.
  - **Pocketsphinx** [11] for testing the model.

#### *Android App*

- Languages we use in coding
  - JAVA
  - XML
  - SQLite Database
- Software
  - Android Studio
  - Sublime Text 3

#### *Microcontroller, Sensor & Other Equipment:*

- Arduino Uno
- Programming Language
  - C
- LED
- Jumper Cables
- Relay
- Breadboard

- Any android phone
- Sensors
  - HC-05 Bluetooth Module

### 3.3 Use Case Modeling and Description

A use case is a methodology used in system analysis to identify, clarify, and organize system requirements [13]. Basically, a use case diagram has emerged on diagrammatic delegation of software elements. Using the use case model, we are capable of a better understanding of possible errors, fault in software processes. Use case model shown in figure 3.3.1.

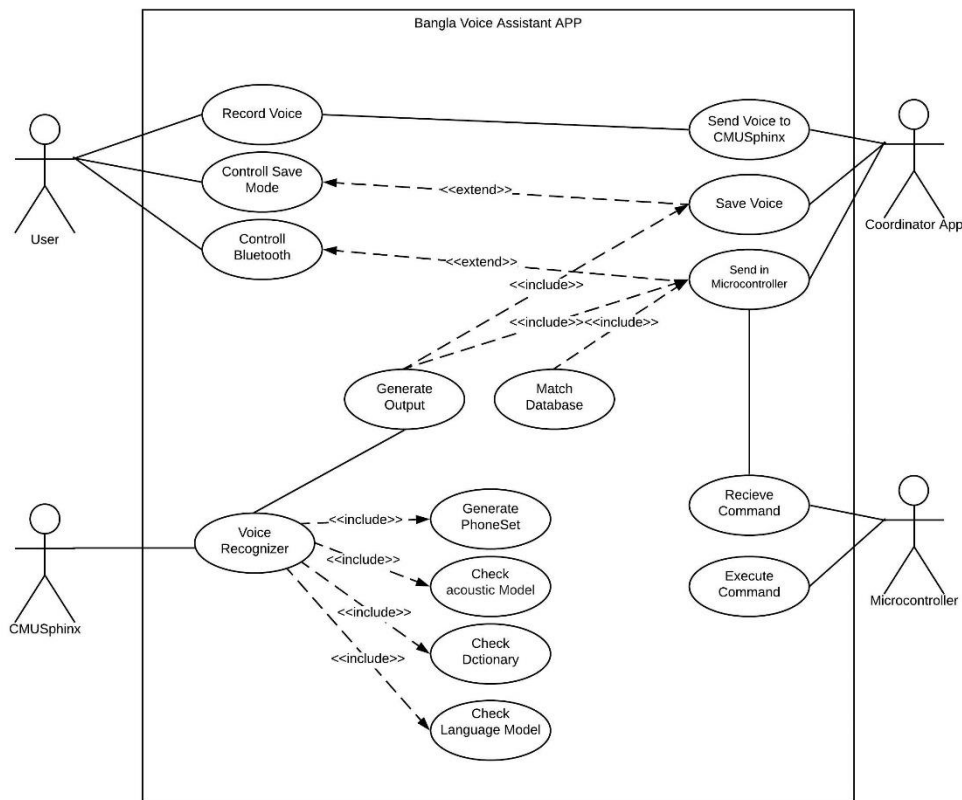


Figure 3.3.1: Use case diagram of Voice Assistant APP

### Use Case Description

There are different use cases used in this diagram design. We are covering some of the most needed use cases. They are:

## Record Voice

Table 1: Use Case Description of Record Voice

<b>Use Case:</b>	<b>Record Voice</b>
<b>Actor:</b>	App User(initiator)
<b>Type:</b>	Primary and essential
<b>Description:</b>	In the beginning, the owner launches the application and perform a voice record. The output of the record then will be displayed on the screen.
<b>Pre-Condition:</b>	Null
<b>Post-Condition:</b>	Null
<b>Use-Cases:</b>	Take voice input from the user.

## Control Bluetooth

Table 2: Use Case Description of Control Bluetooth

<b>Use Case:</b>	<b>Control Bluetooth</b>
<b>Actor:</b>	User(initiator)
<b>Type:</b>	Primary and essential
<b>Description:</b>	Users can control Bluetooth. If the user turns ON/OFF Bluetooth the app will sense and act according to the state. If Bluetooth is off it will prompt the user to turn on the Bluetooth to send data. If the Bluetooth is on it will try to send data to the Microcontroller.
<b>Pre-Condition:</b>	Null
<b>Post-Condition:</b>	Null
<b>Use-Cases:</b>	It helps to turn on/off Bluetooth.

## Recognize voice

Table 3: Use Case Description of Recognize voice

<b>Use Case:</b>	<b>Voice Recognizer</b>
<b>Actor:</b>	System (initiator)
<b>Type:</b>	Primary and essential

<b>Description:</b>	Receives a voice from the user. Then it divides and generates Phone-set. Check the acoustic model and cross-match with the Language model and dictionary. Finally, it generates an output and passes it to the system.
<b>Pre-Condition:</b>	Users must give voice to initiate the process.
<b>Post-Condition:</b>	Null
<b>Use-Cases:</b>	Process Voice input from the user.

### Save Voice

Table 4: Use Case Description of Save Voice

<b>Use Case:</b>	<b>Save Voice</b>
<b>Actor:</b>	System
<b>Type:</b>	Secondary
<b>Description:</b>	After getting processed output from the CMUSphinx the system will check whether the save mood is ON/OFF. If the saving mood is ON the output will be stored along with a pin on the database.
<b>Pre-Condition:</b>	Must have processed data from the CMUSphinx
<b>Post-Condition:</b>	Database access must be granted.
<b>Use-Cases:</b>	Save processed data in the database.

### Send in Microcontroller

Table 5: Use Case Description of Send in Microcontroller

<b>Use Case:</b>	<b>Send in Microcontroller</b>
<b>Actor:</b>	System
<b>Type:</b>	Secondary and essential
<b>Description:</b>	After getting processed output from the CMUSphinx the system will check if the Bluetooth is on or not. If Bluetooth is ON, the system will try to find a match in the database. If a match is found in the database, the PIN no will be sent to the Microcontroller through Bluetooth.
<b>Pre-Condition:</b>	Bluetooth must be ON.

<b>Post-Condition:</b>	There a microcontroller must be connected through Bluetooth to receive data from a mobile device.
<b>Use-Cases:</b>	Send data to microcontroller.

## Execute Command

Table 6: Use Case Description of Recognize voice

<b>Use Case:</b>	<b>Execute Command</b>
<b>Actor:</b>	Microcontroller
<b>Type:</b>	Primary and essential
<b>Description:</b>	Receives command from the connected mobile device and act according to the command.
<b>Pre-Condition:</b>	Must be connected with the mobile device through Bluetooth
<b>Post-Condition:</b>	Home Appliances e.g.: Light, fan must be connected with the equivalent pin.
<b>Use-Cases:</b>	Execute command given from connected devices.

### 3.4 Logical Data Model

The logical data model of our System is shown in figure 3.4.1:

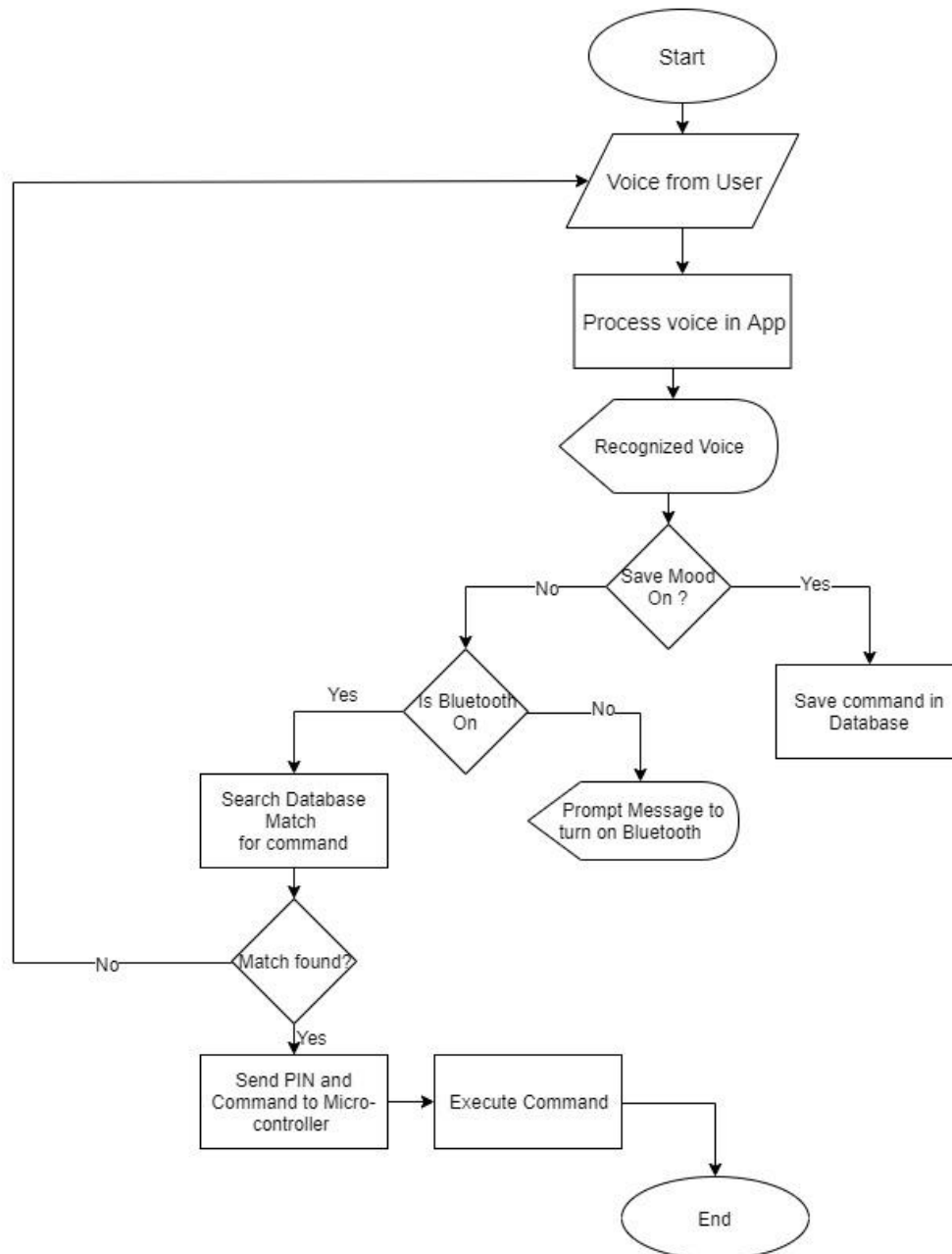


Figure 3.4.1: Full Flowchart of the system

This logical data model shows how the system as a whole works. The model starts form a voice input and ends after successfully executing a command. The process

takes some decision when there's a successful recognition, or the change of Bluetooth state change or voice match found.

Process of Recognizing voice is shown in figure 3.4.2

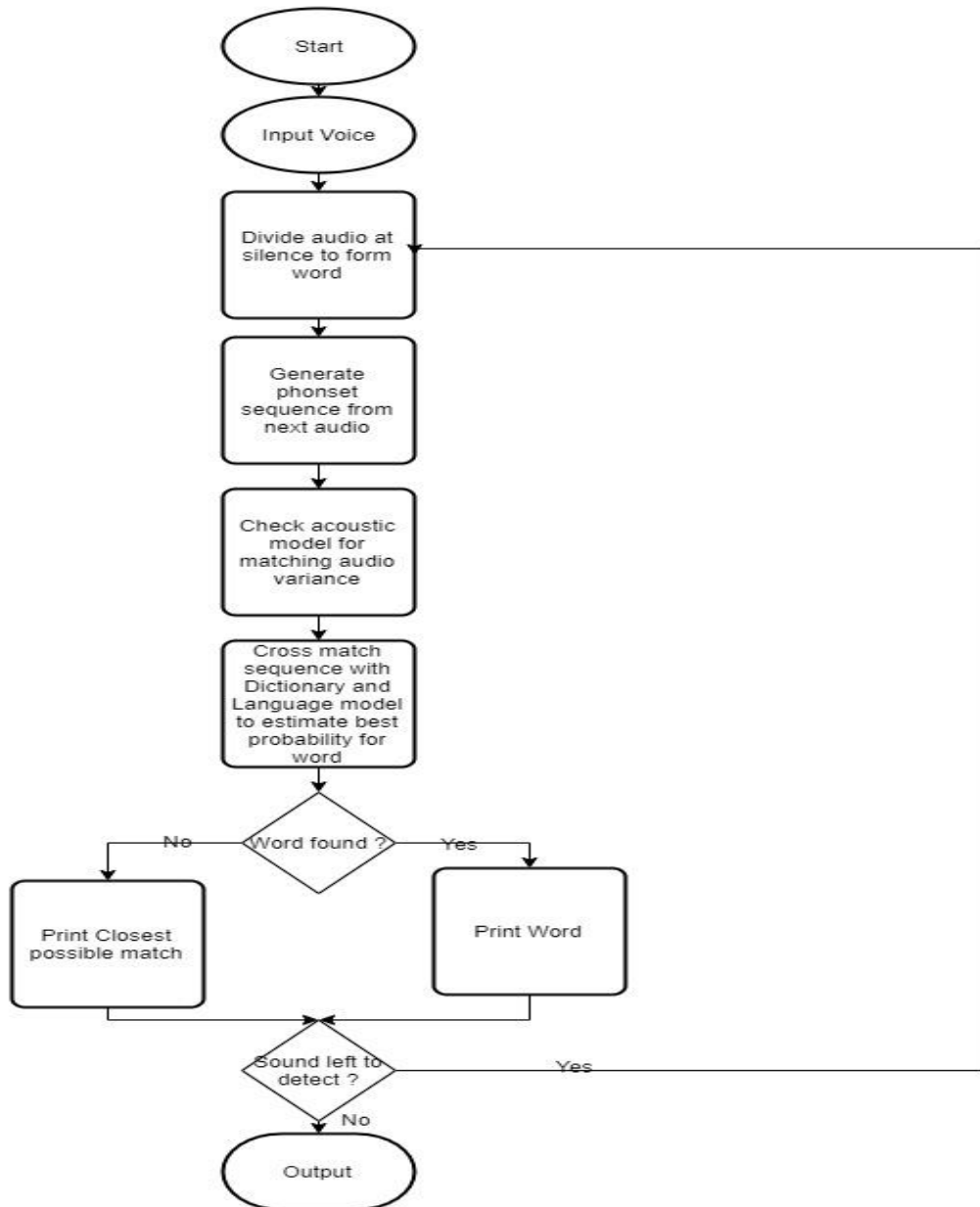


Figure 3.4.2: Flowchart recognizer

Here the total cycle of a speech recognition is shown. The process consists of Language Model, Dictionary and Acoustic models.



Acoustic model training diagram is shown in figure 3.4.3

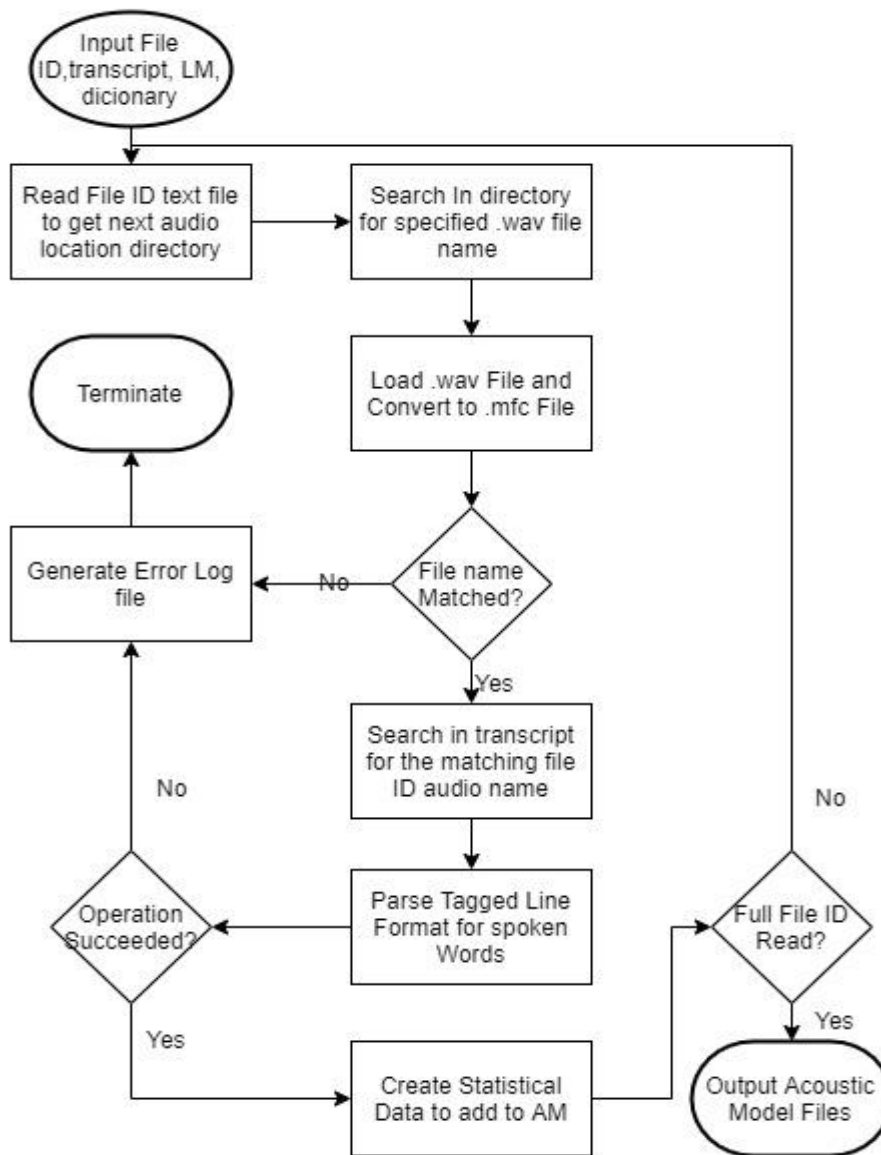


Figure 3.4.3: Acoustic model training diagram

The figure above shows how the acoustic training works. The various steps like configuring the transcription, file IDs audio sources to the final generation is shown here.

### 3.5 Design Requirements

The design requirement of our system has to fulfil user requirements. The smart home assistance comes with a simple application that is user-friendly. Now for designing the system, we focus on the following goals:

**Make user Application Simple and Flexible for the Users:** The home assistance app comes with simple UI design and it is super flexible anyone can use the app easily just saying a voice command

**Efficiency:** the program must be efficient. Since decoders take a lot of resources, we have to design it as efficient as possible. We are running the app on an Android device. So, the resource is limited. We designed the Acoustic model, and Language model as efficient as possible. Enough to get the work done.

**Accuracy:** One of our goals is to make the recognizer accurate. We aim to achieve 80% accuracy in our project.

**Upgradability:** Since the demand of users is changing every day, we aim to design it in such a way that we can upgrade the recognizer, app according to the changing needs of our customers.

## CHAPTER 4

### Design Specification

#### 4.1 Front-end Design

Front-end Design means the visual part of an application. It consists of User Interface and I/O. User can see the changes according to his action. A clean Front-end design can help the user to communicate with the app easily.

So, to have a convenient user experience we tried to make the front-end design as simple as possible. We have used XML (mark-up language) to make the user interface. In the Assistant app, there are basically two activities and one popup menu available. After launching the app the user will come to the main activity page. The design and the code of the page is given on the figure 4.1.1

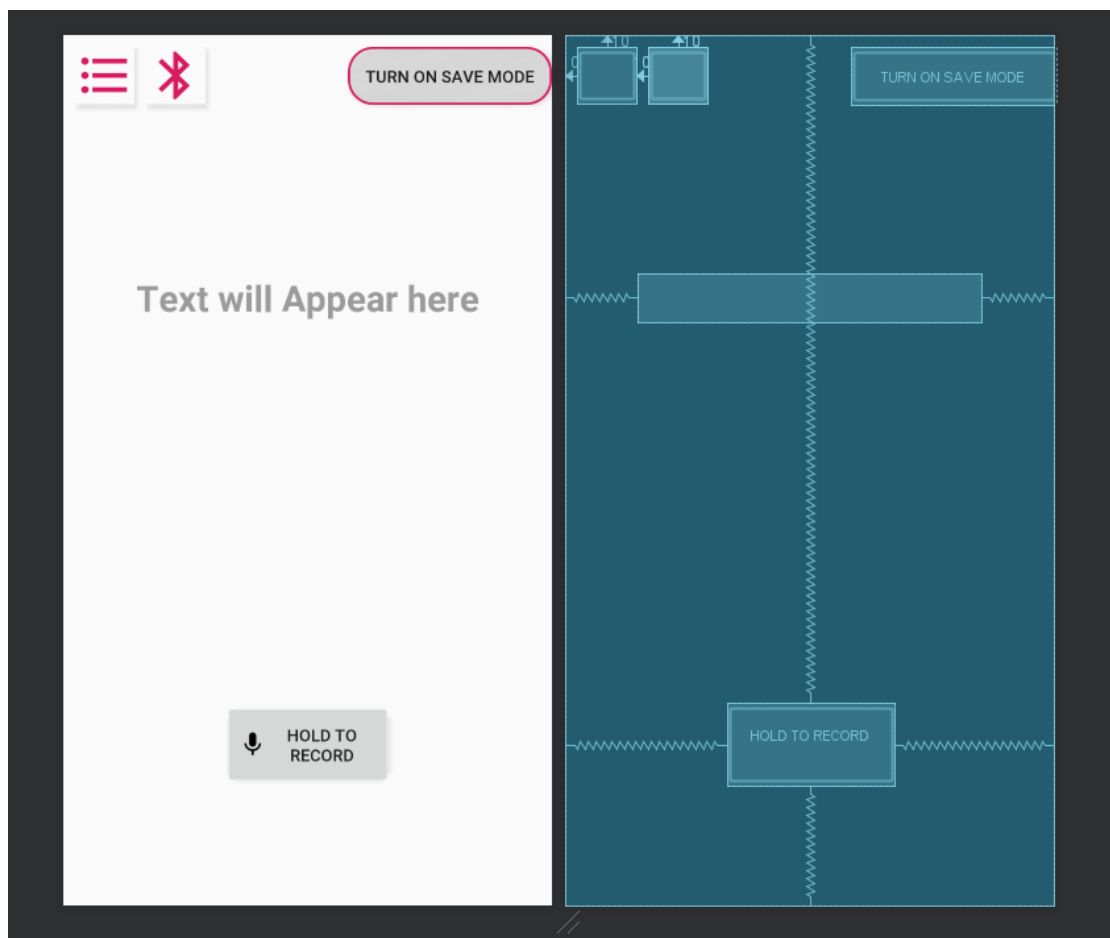


Figure 4.1.1: Stating page design in Android Studio

To have a clean UI and we tried to keep it as simple as possible. Then we have used a RecyclerView to show the already saved command in the app. Here are the design and the XML codes of the command list activity on the figure 4.1.2:

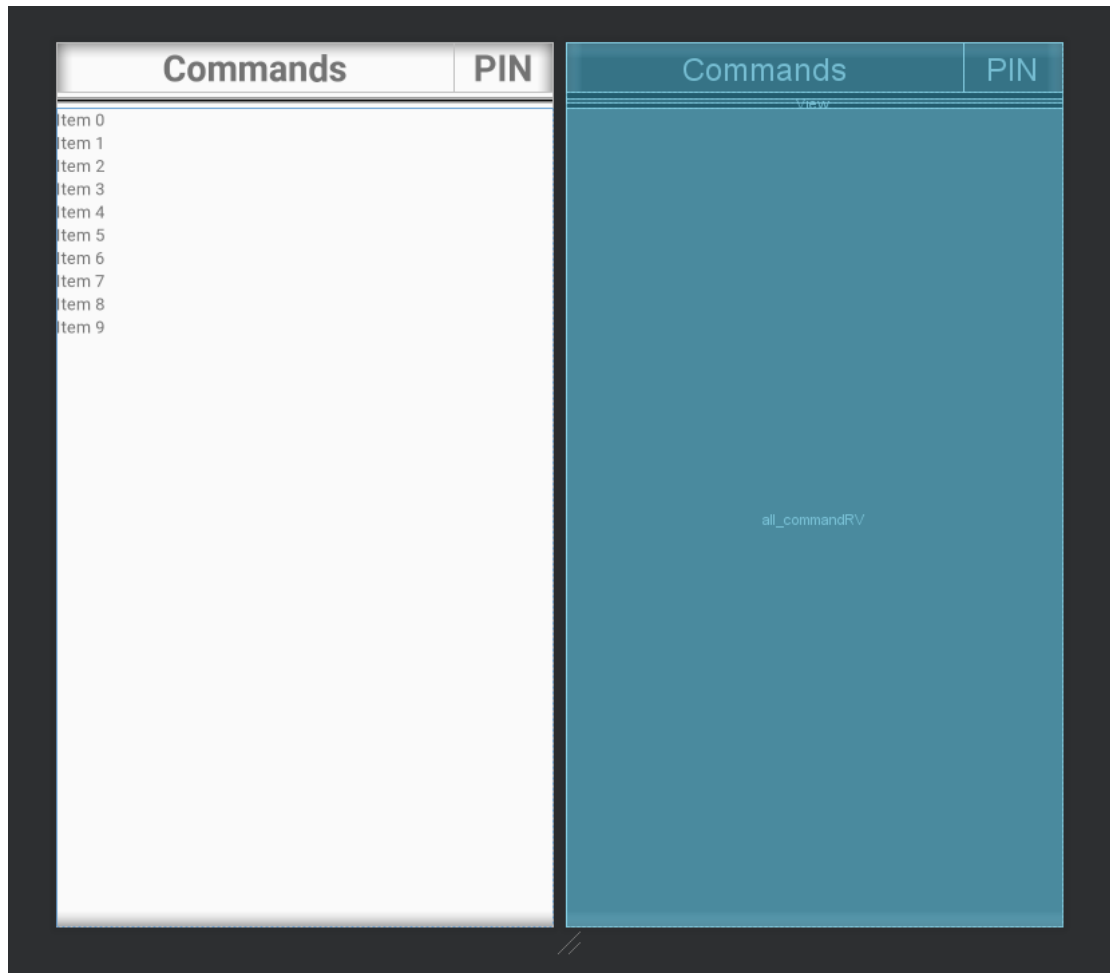


Figure 4.1.2: All list of commands activity design

To save the command we will prompt the user an alert dialogue. The user may select a pin no for the command and save it to the SQLite database. The design and the code are in the figure 4.1.3:

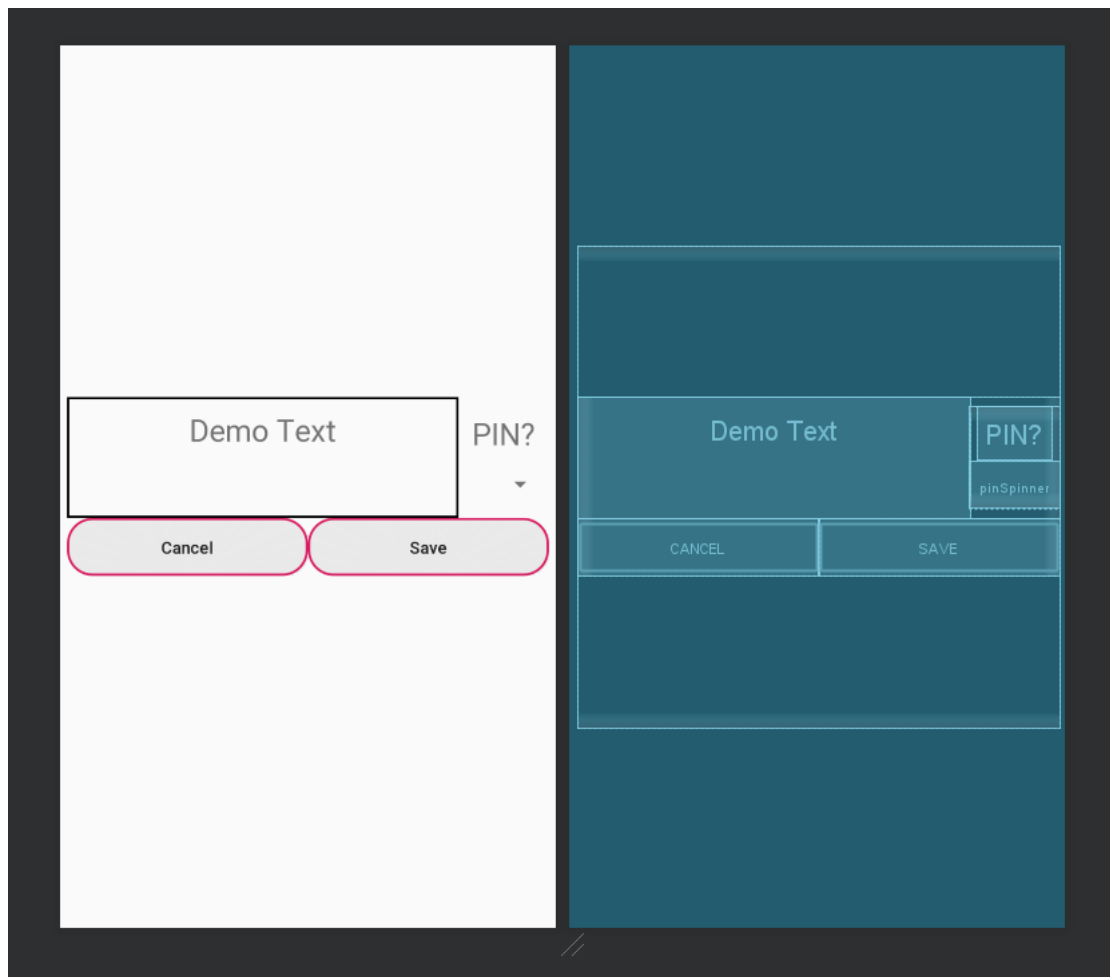


Figure 4.1.3: User prompt to save command

## 4.2 Back-end Design

Generally, the back end refers to indirectly linked devices that respond to end-user activities or requests, e.g., routers, network servers, and e-mail servers [14]. Since our app is an offline app, and it's one of its features, we don't rely on servers and other network equipment and tools. So here we will discuss the things that are happening under the hood. How the recognizer works, how the app handles Bluetooth connection and how the microcontroller handles commands etc.

### 4.2.1 Designing CMU Sphinx Recognizer

CMU Sphinx is the framework we used for building our project's recognizer. So, it's a crucial part of our project. Sphinx provides the necessary tools for building an offline recognizer for any kind of language.

Basically, sphinx takes a real-time voice sample or pre-recorded audio, divides it into phoneset sequences and checks the acoustic model for existing audio variation. Depending on the matches, it generates a closed possible output. The final output is generated in “Banglish” form. Voice to phoneset conversation is given in figure 4.2.1.1

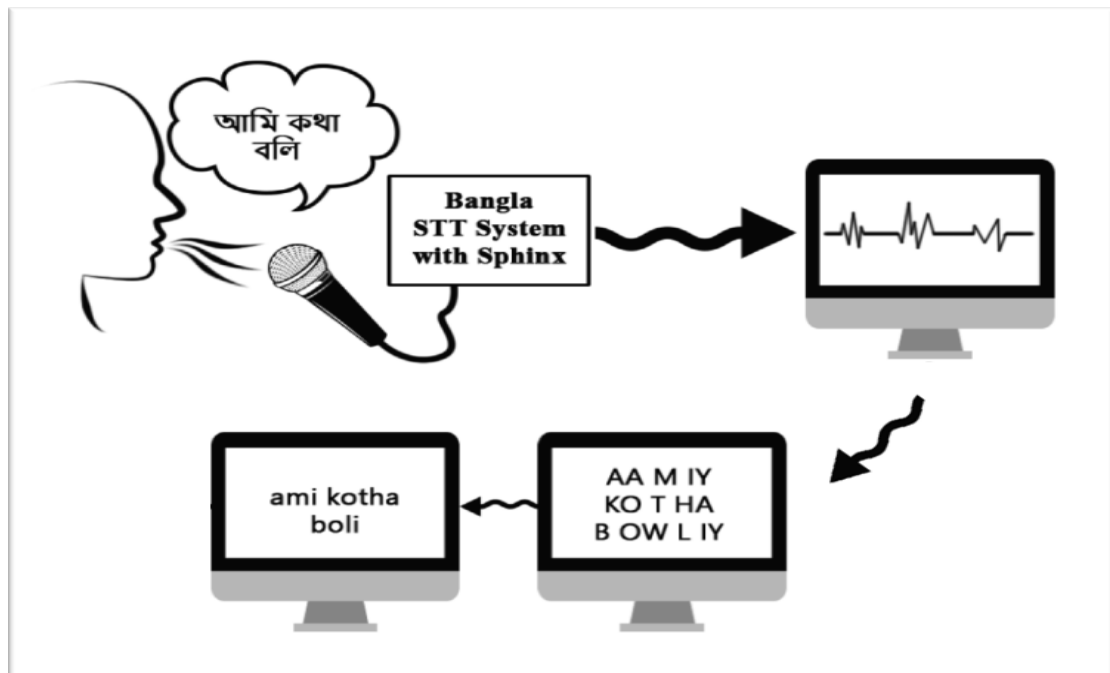


Figure 4.2.1.1: Voice to phoneset conversation [4]

Source: [4]

We need to know a few crucial components of CMU Sphinx. They are

- Language Model
- Phonetic Dictionary
- Acoustic Model

### ***Language Model***

The language model (LM) also known as the statistical model contains probabilistic values for what the next word may be given the recognition of the first. It is created using a tool that has rule sets applied to how a sequence of phone sets may align up based on the trained model and this helps it calculate the statistics of the closest occurrence of a series of utterances detected in a given audio file. Table 7 shows a sample of our language model with its statistical values on the first gram [4].

## Sample Language Model

Table 7: List of language model

Probability	Word
-1.0849	</s>
-1.0849	<s>
-1.9186	AC
-2.3579	BATHROOM
-2.2610	BATI
-1.8139	BEDROOM
-1.7047	BONDHO
-2.9600	CHALU
-2.0569	DINING

### *Building the Language Model*

- Text preparation:

To build a language model, we need to prepare raw text data. So, we gathered all possible commands which a home assistant will need initially. Then we formed meaningful sentences. After that, we used a python3 program for generating the required format for building an ARPA MODEL [17].

```
<s> generally cloudy today with scattered outbreaks of rain and drizzle  
persistent and heavy at times </s>  
<s> some dry intervals also with hazy sunshine especially in eastern parts in  
the morning </s>  
<s> highest temperatures nine to thirteen Celsius in a light or moderate mainly  
east south east breeze </s>  
<s> cloudy damp and misty today with spells of rain and drizzle in most places  
much of this rain will be light and patchy but heavier rain may develop in the  
west later </s>
```

Figure 4.2.1.2: Required format for text2wfreq to work [8]

- We need to generate a vocabulary file from that formatted text with the command

```
text2wfreq < lm.txt | wfreq2vocab > lm.vocab
```

- Then generate an ARPA Language model from that vocabulary file

```
text2idngram -vocab lm.vocab -idngram lm.idngram < lm.closed.txt
idngram2lm -vocab_type 0 -idngram lm.idngram -vocab lm.vocab -arpa
lm.lm
```

- Finally, we need to convert the language model into a binary file:

```
sphinx_lm_convert -i weather.lm -o weather.lm.bin
```

### ***Phonetic Dictionary***

The dictionary file as the name suggests acts as the main lookup source to translate the detected phone set of an audio sample into its consecutive word in Bangla UNICODE. Sphinx samples the input audio by splitting it at the silences and for each split portion, it consults the Acoustic Model to look at past training data and determine what the individual phones may be. After that, it cross-references the detected phones with the dictionary file to see which sequences best match the word list. It may not always find a 100% match for the phone sequence to the word, but this is where probability is used from the language model to determine the closest match [4].

#### Sample Phonetic Dictionary

Table 8: List of Phonetic Dictionary

BONDHO	B AA N D HH OW
CHARO	<i>CH AA R OW</i>
JALAO	<i>JA AH L AA OW</i>
KHULO	K HH Y UW L OW
KORO	K AO R OW
BEDROOM	B EH D R UW M
LIGHT	L AY T
FAN	F AE N



### ***Building the phonetic dictionary***

CMU Sphinx provides a necessary tool **g2p-seq2seq** for building a phonetic dictionary.

```
g2p-seq2seq --interactive --model model_folder_path  
  
> hello  
HH EH L OW
```

It uses a neural network to generate a possible pronunciation for any given word. But our language, Bengali doesn't follow those rules. So, we needed to custom tune those generated phone sets to match the original pronunciations. For example, the word "Bati" generates like this:

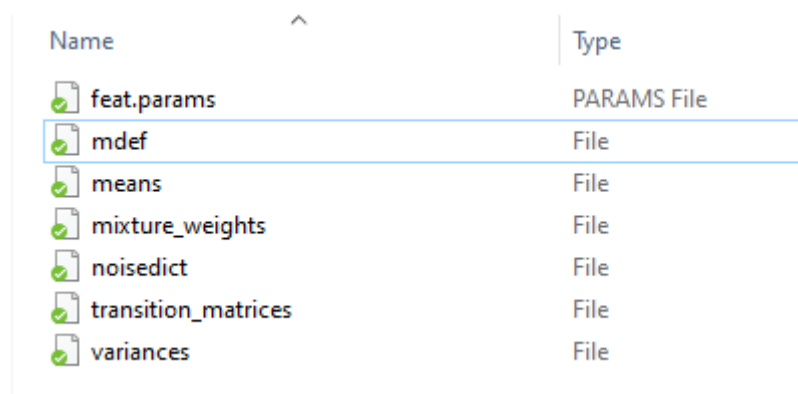
*B AE T IY*

But this pronunciation is wrong. We needed to tune it like:

*B AA T EE*

### ***Acoustic Model***

An acoustic model contains acoustic properties for each senone. There are context-independent models that contain properties (the most probable feature vectors for each phone) and context-dependent ones (built from senones with context) [15]. A sample of an acoustic model is shown in figure 4.2.1.3.



Name	Type
feat.params	PARAMS File
mdef	File
means	File
mixture_weights	File
noisedict	File
transition_matrices	File
variances	File

Figure 4.2.1.3: Final Acoustic Model files.

### ***Generating the Acoustic Model***

Generating the acoustic model is the most important part of all. All the steps below were carefully followed:

- Preparing the directory:

We needed to prepare the following directory (As sphinxtrain requires) [10]

Etc folder contains necessary data for generating the language model. Wav folder contains all the audio files to train the model.

```

├─ etc
|  ├─ project.dic          (Phonetic dictionary)
|  ├─ project.phone      (Phonset file)
|  ├─ project.lm         (Language model)
|  ├─ project.filler     (List of fillers)
|  ├─ project_train.fileids (List of files for training)
|  ├─ project_train.transcription (Transcription for training)
|  ├─ project_test.fileids (List of files for testing)
|  └─ project_test.transcription (Transcription for testing)
└─ wav
   ├─ speaker_1
   |  └─ file_1.wav      (Recording of speech utterance)
   └─ speaker_2
      └─ file_2.wav

```

- Description of present files in etc directory:
  - Project.dic: The main phonetic dictionary file.
  - Project.phone: List of all phones appeared in the dictionary.
  - Project.lm: The language model for the project
  - Project.filler: Fillers like silence, noise etc.
  - Project\_train.fileids: list of all audio files and their names for training.
  - Project\_train.transcription: Transcription of the audio files tagged with their file names for training.
  - Project\_test.fileids: list of all audio files and their names for testing.
  - Project\_test.transcription: Transcription of the audio files tagged with their file names for testing.

- Collected Audio files must be converted into ms wav format with 16bit and 16KHz sampling rate. And audio files must be in mono format.
- Initialize the Training process with the following command

```
sphinxtrain -t project setup
```

- Initializing the training program generated two more files

```
├─ etc
│   └─ feat.params
│   └─ sphinx_train.cfg
```

- Sphinx\_train.cfg contains all the parameters for directing the training process. We tweaked the following:

```
○ $CFG_HMM_TYPE = '.semi.'; # PocketSphinx only
○ $CFG_FINAL_NUM_DENSITIES = 4;
○ $CFG_QUEUE_TYPE = "Queue::POSIX";
○ $CFG_NPART = 5;
```

- Finally, we ran the train with following command

```
sphinxtrain run
```

- Running the train generates following directories in our project directory[10]

```
├─ etc
├─ feat
├─ logdir
├─ model_parameters
├─ model_architecture
├─ result
└─ wav
```

- The output files are generated in model\_parameter folder.

## 4.2.2 Configuring App

### ***Implementing pocketsphinx android***

Pocketsphinx android is a library that provides necessary tools for using language model, dictionary, acoustic model generated by CMU Sphinx. The library is distributed as an Android Archive (AAR) which includes both the binary so files for different architectures and independent Java code [18]. We need to import this library to use the recognizer.

The models and other files are placed in a ‘sync’ directory under ‘src’ folder. The app also needs following permission to work properly

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
```

When the app starts, all the resources from the apk are copied to the internal storage of the device. After that, pocketsphinx uses those files for further recognition.

The recognition happening in three steps.

1. An AsyncTask ‘SetupSphinx’ is initialized. It basically exports the resources from the apk file and sends a variable that contains information about the resource. The method *setupRecognizer* initializes a recognizer variable with the given resources.
2. Then the recognizer starts listening for voice when the start listening is tapped. It calls the recognizer and when the button is released, it stops the recognizer.
3. A set of callback method is called when the recognizer in various states. Like when the speaker is starting to speak *onBeginningOfSpeech()* method is called and when the speaker stops *onEndOfSpeech()* and *onResult()* methods are called. We put a variable in *onResult()* and it collects the final output and uses for further processing

### ***Saving Commands***

To reduce the complexity and speed up the productivity we have implemented a save feature to save the command before sending it to Microcontroller. Users can turn on/off the save mode. If save mode is turned on the system will wait for a sentence output from the CMUSphinx. A save method will be called after releasing the record button.

Then it will look for the output. If it finds any output it will prompt a message to the user and ask the user to enter a pin number for the equivalent command. After selecting a pin when the user hits the save button the command and the pin number will be saved in the SQLite database.

To manage this operation, we have used Room Persistence Library [18]. First, we declared Entities in a model class. Then we created a DAO (Data Access Objects) in a different class. Using that DAO we have called different SQLite operations.

### ***Implementing Bluetooth***

Bluetooth is a technology that communicates wirelessly between two Bluetooth-enabled devices. It uses low energy radio waves to send and receive data. No additional network equipment is required for communication [16]. So, we use Bluetooth technology to communicate with our microcontroller. The implementation of Bluetooth in Android is very interesting. There are basically two parts of communicating with Bluetooth.

- Connecting with another device:

The first thing the program should do is to determine if the Android device supports Bluetooth. Let's say the device supports the Bluetooth. Then we have to check if Bluetooth is enabled or not. If it is not enabled, we have to prompt the user to enable it. After then we have to check the list of paired devices on the existing device. If the pair list is empty, we have to ask the user to manually pair the device we want to communicate. If this is not the case, then we will get the address to Bluetooth address and match it with our existing address in the app. If the address matches we will return a Boolean true from the method.

- Send and Receive data

Let's say we have connected with the device via Bluetooth. Now it's time to dive into the 2<sup>nd</sup> part which is send and receive data. To do so we have to create a socket first and request to create a Service record using UUID.

If everything is connected and well setup, we will then simply call two methods to send and receive data. To listen to any upcoming data from Arduino we will call *beginListenForData()* method. To send data to Arduino we simply call the

*sendCommandToArduino()* method. Finally, if we want to stop Bluetooth connection, we simply close the input stream, output stream, and the socket.

### 4.2.3 Configuring the microcontroller

We have chosen to use Arduino UNO as our microcontroller. It has 14 digital I/O pins and 6 analogue input Pins. It also has SRAM of 2KB and 32KB flash memory. It can provide a 3.3v and 5v power output. Overall, it's a compact powerful embedded device. To communicate with Arduino, we have used an HC-05 Bluetooth module. To power the external components e.g. light, fan, etc. we have used a relay board. In the future, it can be extended up to 12 output at a time.

The basic configuration is shown in figure 4.2.3.1:

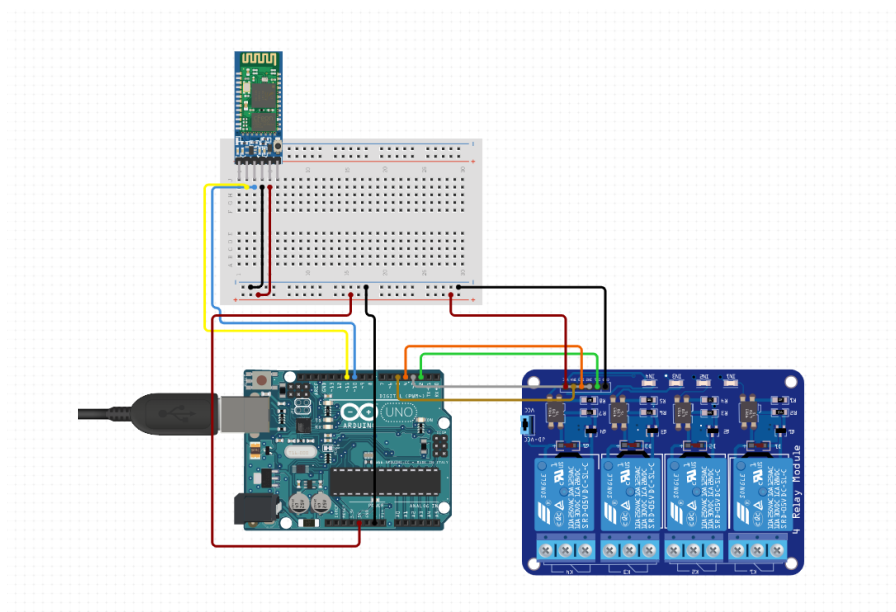


Figure: 4.2.3.1 Arduino connection with Bluetooth and Relay module

Code for receiving data from the Mobile device:

```
1. void setup() {
2.   Serial.begin(9600);
3.   pinMode(2,OUTPUT);
4.   pinMode(3,OUTPUT);
5.   pinMode(4,OUTPUT);
6.   pinMode(5,OUTPUT);
7.   pinMode(6,OUTPUT);
8.   pinMode(7,OUTPUT);
9.   pinMode(8,OUTPUT);
10.  pinMode(9,OUTPUT);
11.  pinMode(10,OUTPUT);
12.  pinMode(11,OUTPUT);
13.  pinMode(12,OUTPUT);
14.  pinMode(13,OUTPUT);
15. }
16.
17. void loop() {
18.   int flag = 0;
19.   char action,pinNo;
20.   int pinNo_int,action_int;
21.   while(Serial.available()){
22.     if(flag == 0){
23.       pinNo = (char)Serial.read();
24.       pinNo_int = (int)pinNo-48;
25.       flag = 1;
26.       Serial.println(pinNo_int);
27.     }
28.     else if(flag == 1){
29.       action = (char)Serial.read();
30.       action_int = action - 48;
31.       Serial.println(action_int);
32.     }
33.   }
34.   digitalWrite(pinNo_int,action_int);
35.   delay(100);
36. }
```

### 4.3 Interaction Design and UX

Users can control their home appliances using Android mobile. There is no need for the internet or any other dependency. The user simply presses a record button and narrate a voice command which he wants to execute. The machine will do the rest. So, to control any home equipment users don't have to do any hard task. Just pressing a button and just ask what he might want to do. We care about our users. So, we keep our app as simple as possible. There is no extra component, or any unnecessary features added to the app.

Users can keep track of the commands saved in the app. If he thinks of changing any command or deleting or updating anything it is also possible. Users can simply

configure Bluetooth simply by pressing a button. Users can also control the save state of the app. All these operations can be easily controlled.

## **4.4 Implementation Requirements**

This system basically works through an android app which can be created with Android Studio. The app uses a voice recognizer which is pocketsphinx. Pocketsphinx is a module of CMU Sphinx. Also, this system uses an Arduino Uno for executing commands sent from the app through Bluetooth.

Here we will discuss the implementation requirements that we needed to complete the project.

### ***Android Studio***

An SDK that enables developers to build, debug, test android applications. It provides source codes, development tools, necessary libraries and an emulator. Applications are written in Java or Kotlin. It also provides tools for Wear OS and Android for IoT. The current Android Studio version is 3.5.

### ***CMU Sphinx***

CMU Sphinx is a framework created by Carnegie Mellon University to provide necessary tools for building an offline speech recognizer. Pocket sphinx is a tool that is the simplest version of the decoder for providing the speech to text output at minimal resource usage. It has also an android module that works with any android device.

### ***Arduino Uno***

Arduino is a company that manufactures open-source single board hardware as microcontrollers and software [20]. Arduino provides easy and efficient ways to execute commands written in C language. We used the Arduino Uno, which is a cheap and affordable product. It has 1KB of EPROM, 2KB of SRAM and 32KB of flash memory. In its heart, there's an ATmega328P chip clocked at 16 MHz. It contains 6 analogue and 14 digital pins.



## CHAPTER 5

### Implementation and Testing

#### 5.1 Implementation of Database

For the database we have used SQLite database. We have stored a pin number of the equivalent command given by the user. After completing the speech of the user we will check if that matched with the database. If it matched with the database we check the trigger command and then send the pin and trigger command to the Arduino.

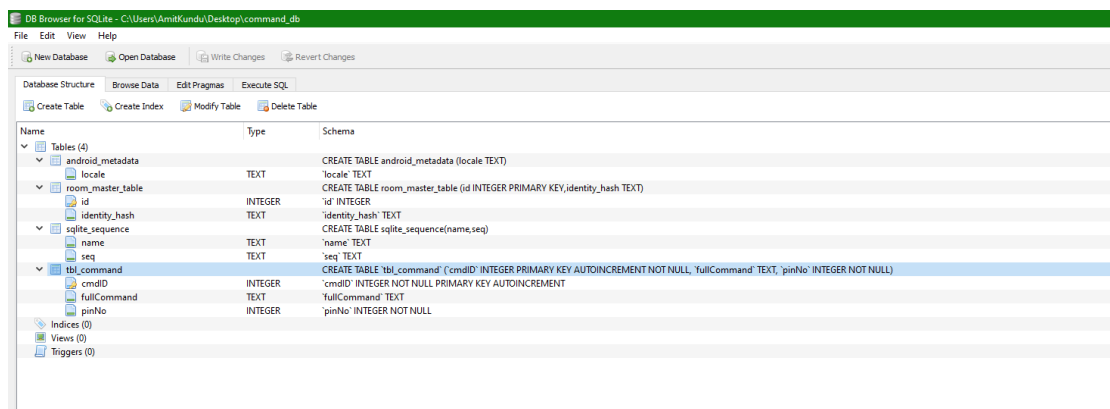


Figure 5.1.1: Implementation of database

#### 5.2 Implementation of Front-end Design

Front-end design plays a vital role; the usages of an application depend on how convenient the design is. The effectiveness of an application depends on the function as well as in its design. If the interface is easy to use the user may find it attractive to him. So here we discuss the front-end design implementations. The lurching page is shown in the figure 5.2.1:



Figure 5.2.1: Front-end design

In this activity, the user can record voice. Turn on/off Bluetooth. And turn on/off save mode.

If the user wants to save any command a popup window will open and ask for the pin number. The design is shown on the figure 5.2.2:

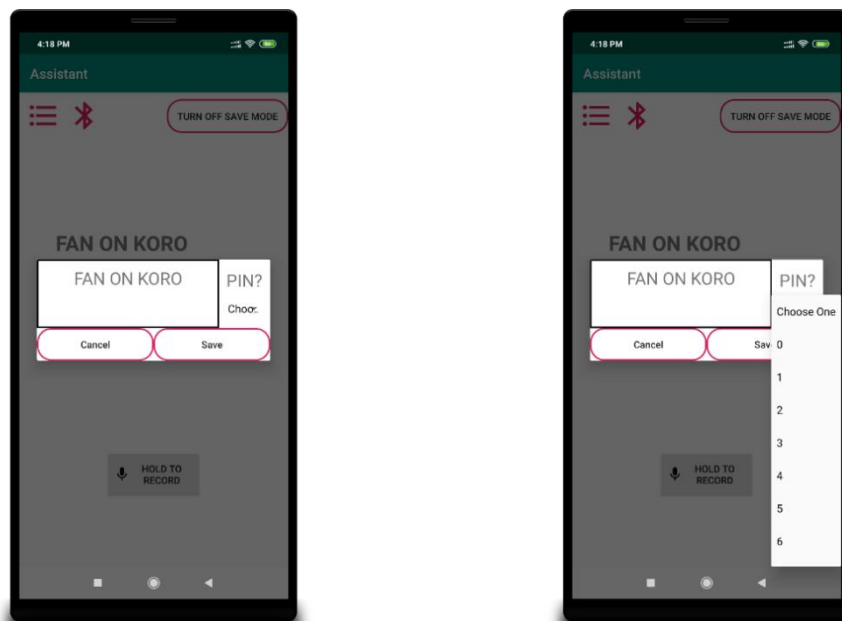


Figure 5.2.2: Saving Command popup window

Users can check the saved command list also by tapping the list button on the left top of the lunching activity. It is shown in figure 5.2.3:



Figure 5.2.3: Command list

### 5.3 Implementation of Interactions

Defining the right interaction model is a foundational requirement for any digital system and contributes to a cohesive, overall UX architecture [19]. The more an application is interactive the more it is used [12]. We have tried to make the application interactive to the user so that the user has to put less effort into a single operation. So when the user record a voice the voice command is sent to the Arduino and the Arduino will execute the command according to user demand. Users can also save the command for defining pin number of home appliances on which PIN they are connected.

### 5.4 Testing Implementation

If we want a bug free system, there is no alternative to testing. So, we have to follow some testing procedures to check whether the system is stable or not. Here is the description of our test:

ID	Test Case Description	Test Case Procedure	Expected Output	Test date	Result
TC1	Checking Bluetooth Connectivity	1. Turn on Bluetooth of the system 2.Pair with HC-05 module	App should be able to connect with the HC-06 module	2-Nov-19	Passed

		3. Click on the Bluetooth icon of the App			
TC2	Checking Save mode	1.After launching the app Tab on the save mode Button 2.When recording ends a popup should come and ask a pin number. 3. After saving the pin the popup window should go away.	Should be able to store commands in the database	2-Nov-19	Passed
TC3	Checking command list	1.After launching the app click on the command list.	Should be able to see the saved commands	2-Nov-19	Passed
TC4	Real-time voice data transmission	1.After launching the app click on the Bluetooth connectivity button. 2.After connected with Bluetooth click on the record button. 3. After releasing the button press, the output should be displayed on the screen. 4. The data should be sent on Arduino via Bluetooth.	Data should arrive on Arduino	2-Nov-19	Passed
TC5		1.Testing Arduino code implementation	All the commands should be executed correctly.	2-Nov-19	Passed

## 5.5 Test Results and Reports

Result summary on table 9

Table 9: Result summary

<b>System Nam :</b>	Bangla Home Assistance		
Pass	5	Pending	0
Fail	0	The number of test cases:	5

The system performance was very near to the expected result. We have successfully created an offline Bangla voice recognizing home automation.

## CHAPTER 6

### Conclusion and Future Scope

#### 6.1 Discussion and Conclusion

The smart home lets you control a range of connected devices in your home from a smartphone [1]. By using your smartphone, you just control your home appliances in one hand just saying a Bengali voice command. In this home assistant, we use the CMUSphinx framework for voice recognition and develop a mobile application to recognize the Bengali voice commands. Our app also connects with a microcontroller via Bluetooth connection and it executes many actions. We used Audacity which is a digital audio workstation to manipulate the recorded voice data.

#### 6.2 Scope for Further Developments

Our project is far from a complete state. We tried to show that it is possible to create a home assistant device that uses Bengali. But it's not in its final state. We have a long way ahead. We plan to create custom hardware for this system and make it simpler to implement. Also, recognition is pretty limited right now. We plan to improve the recognition and add support for as many words as possible. Bengali is a complicated language. It doesn't follow rules made for other English or similar languages, so automation is hard. Hand creating every rule is not possible for a small team like ours. So, we plan to increase the manpower and generate a bigger and better recognizer for Bengali.

## Appendices

### Appendix A: Project Reflection

From the Fall-2018 semester, we started our journey to making a system that can ease our daily life. We decided to make a home assistant that can make home automation easily for everyone without any effort. Saying a voice command our system will execute many tasks like turning on or off lights, fan, and many electronic appliances. And the system will work with our mother tongue Bengali language. In the beginning, we create a Bengali language model and train the voice data. We collect voice data about around 100 people. After train the voice data finally, we reach our goal. About 85% accuracy we get from our model. We also create an app that takes voice command. A user can modify, edit and delete any tasks from the app. In the future, we will Improve the recognizer by implementing an algorithm for the closest matches.

### Appendix B: Related Diagrams

Figure: 4.1.1 Stating page design in Android Studio

Figure: 4.1.2 All list of commands activity design

Figure: 4.1.3 User prompt to save command

Figure: 4.2.1.1 Voice to phone set conversation Source: [4]

Figure4.2.1.2: Required format for text2wfreq to work [8].

Figure 4.2.1.3: Final Acoustic Model files.

## References

- [1] "A guide to home automation and smart home assistants - Clickatell", *Clickatell*, 2019. [Online]. Available: <https://www.clickatell.com/articles/technology/home-automation-smart-home-assistants/#targetText=The%20smart%20home%20lets%20you,your%20home%20from%20your%20smartphone.&targetText=Just%20as%20there%20are%20virtual,aid%20home%20automation%20make%20sense>. [Accessed: 26- Oct- 2019].
- [2] "Languages of Bangladesh", *En.wikipedia.org*, 2019. [Online]. Available: [https://en.wikipedia.org/wiki/Languages\\_of\\_Bangladesh#targetText=The%20national%20language%20C%20Bengali%20is,monolingual%20country%20in%20South%20Asia](https://en.wikipedia.org/wiki/Languages_of_Bangladesh#targetText=The%20national%20language%20C%20Bengali%20is,monolingual%20country%20in%20South%20Asia). [Accessed: 26- Oct- 2019].
- [3] N. Shmyrev, "CMUSphinx Open Source Speech Recognition", *CMUSphinx Open Source Speech Recognition*, 2019. [Online]. Available: <https://cmusphinx.github.io/>. [Accessed: 27- Oct- 2019].
- [4] H. Kabir, R. Ahmed and A. Nasib, "Real time bengali speech to text conversion using CMU sphinx", *Dspace.bracu.ac.bd*, 2019. [Online]. Available: <http://dspace.bracu.ac.bd/xmlui/handle/10361/9546>. [Accessed: 01- Nov- 2019].
- [5] N. Shmyrev, "Building an application with sphinx4", *CMUSphinx Open Source Speech Recognition*, 2019. [Online]. Available: <https://cmusphinx.github.io/wiki/tutorialsphinx4/>. [Accessed: 27- Nov- 2019].
- [6] S. Rabby, M. Faisal and M. Islam, "Web Application Based Wireless Home Automation System", *Dspace.daffodilvarsity.edu.bd*, 2019. [Online]. Available: <http://dspace.daffodilvarsity.edu.bd:8080/handle/123456789/3356>. [Accessed: 27- Oct- 2019].
- [7] D. Kumar, S. Singh and N. Sinha, "Home Automation via Bluetooth using Android Application", *ABHIYANTRIKI: ABHIYANTRIKI: An International Journal of Engineering & Technology*, vol. 4, 4, no. 2394-627, 2017. [Accessed 27 October 2019].
- [8] N. Shmyrev, "Building a language model", *CMUSphinx Open Source Speech Recognition*, 2019. [Online]. Available: <https://cmusphinx.github.io/wiki/tutoriallm/>. [Accessed: 28- Oct- 2019].
- [9] "cmusphinx/g2p-seq2seq", *GitHub*, 2019. [Online]. Available: <https://github.com/cmusphinx/g2p-seq2seq>. [Accessed: 28- Oct- 2019].
- [10] N. Shmyrev, "Training an acoustic model for CMUSphinx", *CMUSphinx Open Source Speech Recognition*, 2019. [Online]. Available: <https://cmusphinx.github.io/wiki/tutorialam/>. [Accessed: 28- Nov- 2019].
- [11] N. Shmyrev, "Building an application with PocketSphinx", *CMUSphinx Open Source Speech Recognition*, 2019. [Online]. Available: <https://cmusphinx.github.io/wiki/tutorialpocketsphinx/>. [Accessed: 28- Oct- 2019].
- [12] N. TABASSUM, "TUTORS MAP: AN ANDROID APP FOR PRIVATE TUTOR", *Dspace.library.daffodilvarsity.edu.bd*, 2019. [Online]. Available: <http://dspace.library.daffodilvarsity.edu.bd:8080/handle/20.500.11948/2717>. [Accessed: 29- Oct- 2019].
- [13] "What is use case diagram (UML use case diagram)? - Definition from WhatIs.com", *WhatIs.com*, 2019. [Online]. Available: <https://whatis.techtarget.com/definition/use-case-diagram>. [Accessed: 30- Oct- 2019].
- [14] "What is Front and Back Ends? - Definition from Techopedia", *Techopedia.com*, 2019. [Online]. Available: <https://www.techopedia.com/definition/24794/front-and-back-ends#targetText=Front%20and%20back%20ends%20refer,computer%20hardware%20and%20user%20layers.&targetText=The%20back%20end%20refers%20to,servers%20and%20e%2Dmail%20servers>.



[Accessed: 30- Oct- 2019].

[15] N. Shmyrev, "Basic concepts of speech recognition", *CMUSphinx Open Source Speech Recognition*, 2019. [Online]. Available: <https://cmusphinx.github.io/wiki/tutorialconcepts/>. [Accessed: 30- Oct- 2019].

[16] "What is Bluetooth Technology Used for?", *Smallbusiness.chron.com*, 2019. [Online]. Available: <https://smallbusiness.chron.com/bluetooth-technology-used-for-58102.html>. [Accessed: 30- Oct- 2019].

[17] N. Shmyrev, "ARPA Language models", *CMUSphinx Open Source Speech Recognition*, 2019. [Online]. Available: <https://cmusphinx.github.io/wiki/arpaformat/>. [Accessed: 31- Oct- 2019].

[18] N. Shmyrev, "PocketSphinx on Android", *CMUSphinx Open Source Speech Recognition*, 2019. [Online]. Available: [https://cmusphinx.github.io/wiki/tutorialandroid/?fbclid=IwAR0otLYZqUXzYgUhX0V\\_kcBPHEdQ12M6w1m2wbe10a7tui\\_5Q4pnY2RIblk](https://cmusphinx.github.io/wiki/tutorialandroid/?fbclid=IwAR0otLYZqUXzYgUhX0V_kcBPHEdQ12M6w1m2wbe10a7tui_5Q4pnY2RIblk). [Accessed: 31- Oct- 2019].

[19] D. Design, "Defining an Interaction Model: The Cornerstone of Application Design :: UXmatters", *Uxmatters.com*, 2019. [Online]. Available: <https://www.uxmatters.com/mt/archives/2012/01/defining-an-interaction-model-the-cornerstone-of-application-design.php>. [Accessed: 01- Nov- 2019].

[20] "Arduino", *En.wikipedia.org*, 2019. [Online]. Available: <https://en.wikipedia.org/wiki/Arduino?fbclid=IwAR13sVtRmy4bPclAsQVr1HI8Qr0tF-97FKeCEPXArJsR3yQsvMmLE-gMovY>. [Accessed: 02- Nov- 2019].

# BANGLA HOME ASSISTANT

---

## ORIGINALITY REPORT

---

13%

SIMILARITY INDEX

9%

INTERNET SOURCES

1%

PUBLICATIONS

11%

STUDENT PAPERS

---

## PRIMARY SOURCES

---

1

Submitted to Daffodil International University

Student Paper

4%

---

2

[cmusphinx.github.io](https://cmusphinx.github.io)

Internet Source

1%

---

3

[dspace.daffodilvarsity.edu.bd:8080](https://dspace.daffodilvarsity.edu.bd:8080)

Internet Source

1%

---

4

[ama2595.blogspot.de](https://ama2595.blogspot.de)

Internet Source

1%

---

5

[hsbp.org](https://hsbp.org)

Internet Source

1%

---

6

Burhanuddin Lakdawala, Farhan Khan, Arif Khan, Yash Tomar, Rahul Gupta, Ashfaq Shaikh. "Voice to Text transcription using CMU Sphinx A mobile application for healthcare organization", 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), 2018

Publication

1%

---

7

Submitted to University of Arizona

---

Student Paper

<1 %

---

8

Submitted to University of Hertfordshire

Student Paper

<1 %

---

9

[www.techopedia.com](http://www.techopedia.com)

Internet Source

<1 %

---

10

[www.uxmatters.com](http://www.uxmatters.com)

Internet Source

<1 %

---

11

[searchsoftwarequality.techtarget.com](http://searchsoftwarequality.techtarget.com)

Internet Source

<1 %

---

12

Submitted to The University of Manchester

Student Paper

<1 %

---

13

[www.techilatechnologies.com](http://www.techilatechnologies.com)

Internet Source

<1 %

---

14

Submitted to University of Greenwich

Student Paper

<1 %

---

15

[doowop-net.com](http://doowop-net.com)

Internet Source

<1 %

---

16

Submitted to Western Mindanao State  
University

Student Paper

<1 %

---

17

Submitted to Asia Pacific Institute of  
Information Technology

Student Paper

<1 %

---

18	<a href="https://hdl.handle.net">hdl.handle.net</a> Internet Source	<1 %
19	<a href="https://amsdottorato.unibo.it">amsdottorato.unibo.it</a> Internet Source	<1 %
20	<a href="https://dspace.bracu.ac.bd">dspace.bracu.ac.bd</a> Internet Source	<1 %
21	Submitted to University of Central England in Birmingham Student Paper	<1 %
22	<a href="http://www.scss.tcd.ie">www.scss.tcd.ie</a> Internet Source	<1 %
23	Submitted to Dubuque Comm School District Student Paper	<1 %
24	<a href="http://www.actappraisal.com">www.actappraisal.com</a> Internet Source	<1 %
25	<a href="http://alexyang.net">alexyang.net</a> Internet Source	<1 %
26	<a href="http://doras.dcu.ie">doras.dcu.ie</a> Internet Source	<1 %
27	Submitted to University of Sheffield Student Paper	<1 %
28	Submitted to Majan College Student Paper	<1 %

---

29	<b>Submitted to London School of Commerce</b> Student Paper	<1%
30	<b>Submitted to Coventry University</b> Student Paper	<1%
31	<b>Submitted to Glasgow Caledonian University</b> Student Paper	<1%
32	<b>Submitted to University Tun Hussein Onn Malaysia</b> Student Paper	<1%

---

---

Exclude quotes      Off

Exclude matches      Off

Exclude bibliography      On