APPENDIX A

SAMPLE SCREEN SHOTS

To start with, fig A1 represents animated representation of an Air Traffic Controller. In existing system he is responsible for safety maneuvering of aircraft.

Fig A1 Air Traffic Controller

Courtesy: VectorStock

The standard method of communication between an air traffic controller and a pilot is voice radio, using either Very High Frequency (VHF) bands for line-of-sight communication or High Frequency (HF) bands for long-distance communication (such as that provided by Shanwick Oceanic Control). Fig A2 represents voice radio communication.

Fig A2 Voice radio Communication

Courtesy: Arabian Airspace

The busy controller in main control tower receives the request from the pilot and transfers the response. Fig A3 shows the interior working of ATC

Fig A3 ATC Interior

Courtesy: Reddit r/Pics

The main hardware components that we are going to use are shown in fig A4. A good performing Personal computer with internet connectivity and earphones for passing the input.

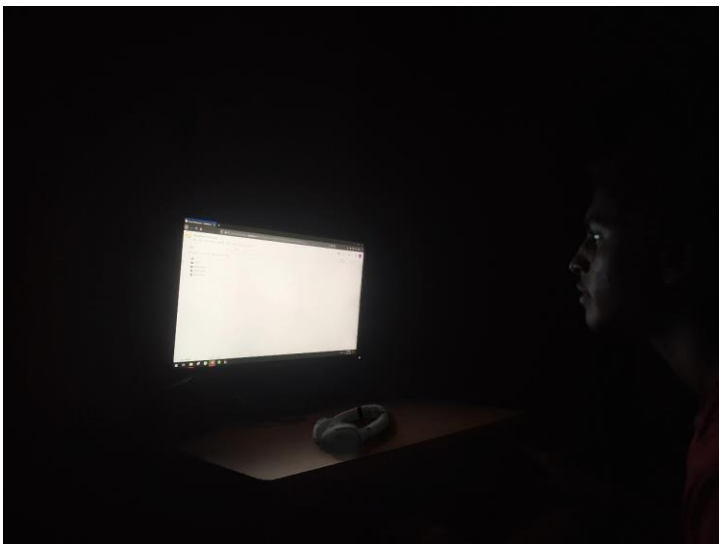

Fig A4 PC and earphones

Initially the audio (Speech request) has to be passed as input the system and the system responds after receiving the audio as shown in fig A5 and A6.

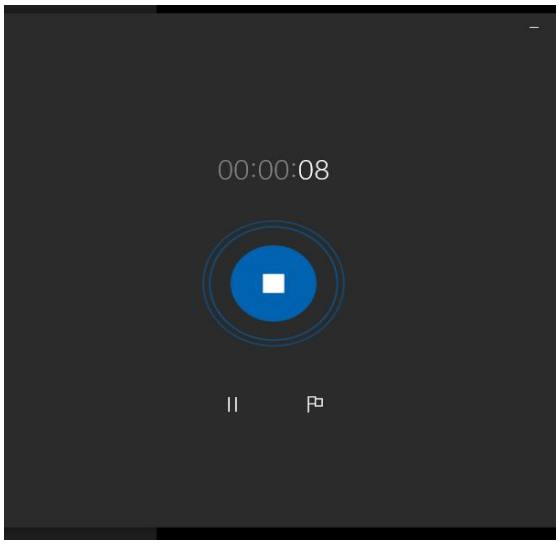

Fig A5 Audio input

```
r=sr.Recognizer()
harvard=sr.AudioFile(fn)
with harvard as source:
  audio = r.record(source)
if(audio):
  print("Audio Received Successfully")
else:
  print("Audio Error")
```

```
Audio Received Successfully
```

Fig A6 System Response after receiving audio

The first module of our system converts the pilot's request speech to text. This module can also be called as Speech recognition module. This is a vital task which takes place initially in the entire process.. Then the collected audio file is converted to text, by including google speech to text package in the source code and relevant text is obtained. This is done so, in order to understand the pilot's need and intentions. The audio converted to text is shown in the below fig A7

```
[7] from google.colab import files
    uploaded= files.upload()

    for fn in uploaded.keys():
      print('User uploaded file "{name}" with length {length} bytes'.format(
          name=fn, length=len(uploaded[fn])))
```

```
Browse... audio_1.wav
audio_1.wav(audio/wav) - 1075278 bytes, last modified: n/a - 100% done
Saving audio_1.wav to audio_1.wav
User uploaded file "audio_1.wav" with length 1075278 bytes
```

```
[8] harvard=sr.AudioFile('audio_1.wav')
    with harvard as source:
      audio = r.record(source)
```

```
audio_text=r.recognize_google(audio)
print(audio_text)
```

```
Air India 101 poling short of Runway ready for takeoff
```

Fig A7 Audio converted to text

Figure A8 is the screenshot of 2nd module, where the LSTM working takes place.

```
[ ] import nltk
    nltk.download('stopwords')
    nltk.download('punkt')
    from nltk.corpus import stopwords
    from nltk.tokenize import word_tokenize
    stop_words = set(stopwords.words('english'))
    word_tokens = word_tokenize(audio_text)

    filtered_sentence = [w for w in word_tokens if not w in stop_words]

    filtered_sentence = []

    for w in word_tokens:
      if w not in stop_words:
        filtered_sentence.append(w)

    print(word_tokens)
    print(filtered_sentence)
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
['Air', 'India', '101', 'poling', 'short', 'of', 'Runway', 'ready', 'for', 'takeoff']
['Air', 'India', '101', 'poling', 'short', 'Runway', 'ready', 'takeoff']
```

Fig A8 Text Summarization

Figure A9 represents the screenshot of generating suitable response from CSV after the summarization takes place.

```
import pandas
with open('response.csv', mode='r') as csv_file:
    csv_reader = csv.reader(csv_file,delimiter=',')
    request=[]
    response=[]
    actual_response=''
    i=0
    for row in csv_reader:
     request.append(str(row[0]))
     response.append(str(row[1]))
    for i in range(len(filtered_sentence)):
      for k in range(len(request)):
          if(filtered_sentence[i]==request[k]):
              actual_response=response[k]
file = open("airlines.txt",'r')
tmp=' '
count=0
while True:
count+=1
line=file.readline().replace('\n','.')
tmp=line
if (tmp in audio_text):
   found_flag=1
   print(actual_response+' '+tmp+'101')
```

```
Cleared for Takeoff  AirIndia 101
```

Fig A9 Response Generation

Fig A10 shows the audio response.

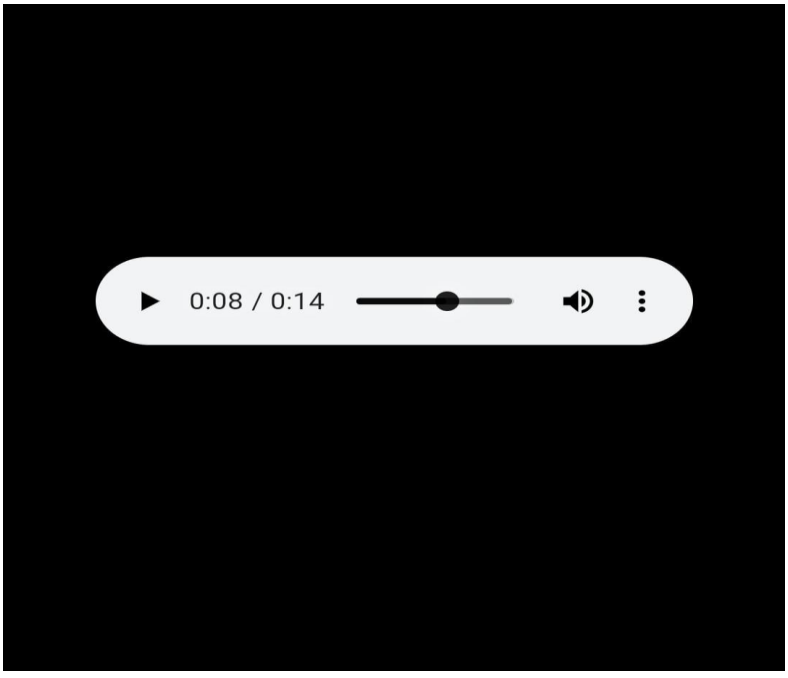

Fig A10 Audio response

APPENDIX B
SAMPLE CODING

```
pip install SpeechRecognition #Speechrecognition module
pip install google-cloud-speech #GoogleCloudSpeech Module
pip install apiai #another module
pip install nltk
pip install pandas
import speech_recognition as sr
import nltk
import pandas
```

#audio receiver function

```
r=sr.Recognizer()
harvard=sr.AudioFile(fn)
with harvard as source:
audio = r.record(source)
if(audio):
        print("Audio Received Successfully")
else:
        print("Audio Error")
```

#recognizer function for speech2text

```
audio_text=r.recognize_google(audio)
print(audio_text)
```

```
#tokenization module
nltk.download('stopwords')
nltk.download('punkt')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
stop_words = set(stopwords.words('english'))
word_tokens = word_tokenize(audio_text)
filtered_sentence = [w for w in word_tokens if not w in stop_words]
filtered_sentence = []
for w in word_tokens:
        if w not in stop_words:
                filtered_sentence.append(w)
print(word_tokens)
print(filtered_sentence)
#end of tokenization module

#response module
with open('response.csv', mode='r') as csv_file:
csv_reader = csv.reader(csv_file,delimiter=',')
request=[]
response=[]
actual_response=''
i=0
for row in csv_reader:
 request.append(str(row[0]))
 response.append(str(row[1]))
for i in range(len(filtered_sentence)):
```

```
for k in range(len(request)):
    if(filtered_sentence[i]==request[k]):
        actual_response=response[k]
    file = open("airlines.txt",'r')
    tmp=' '
    count=0
    while True:
    count+=1
    line=file.readline().replace('\n','.')
    tmp=line
    if (tmp in audio_text):
            found_flag=1
    print(actual_response+' '+tmp+'101')
    #end of response module
```

APPENDIX C

SYSTEM REQUIREMENTS

HARDWARE SPECIFICATION

- 4 GB RAM or Higher
- 256 GB Storage or higher
- Internet Connectivity
- Recording Microphones
- Basic Computer Peripherals

SOFTWARE SPECIFICATION

- Windows 7 or Higher
- Google Colab
- NLTK
- Keras Library
- Google Cloud Speech
- Pandas Library
- NumPy Library
- Speech Recognition library
- APIAI Library

APPENDIX D

PUBLICATIONS

[1] Kailasa Eswaran I, Balaji C and M Ashok, presented a paper titled "Implementation of LSTM in ATC for Avionics" in the "8th International Conference on Contemporary Engineering and Technology 2020 (ICCET'2020)" organized by Prince Shri Venkateshwara Padmavathy Engineering College on 14th and 15th March 2020.