INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 4, Issue 3)
Available online at: www.ijariit.com

GIF – Get IT Filled

Aiswarya ManojKumar
aiswaryamanoj.manoj3@gmail.com
Vimal Jyothi Engineering College, Chemperi, Kerala

Anagha Pallen Pavithran
anagha444@gmail.com
Vimal Jyothi Engineering College, Chemperi, Kerala

Bagya Girish
bhagya.g6@gmail.com
Vimal Jyothi Engineering College, Chemperi, Kerala

Sneha C K
snehasudheerck@gmail.com
Vimal Jyothi Engineering College, Chemperi, Kerala

ABSTRACT

GIF - Get It Filled is an intelligent system that automatically generates the deposit and withdrawal forms required in a bank. In the current scenario, the old-aged & illiterate customers require the help of literate people to fill a deposit or withdrawal form in a bank. To help improve such scenarios, we are introducing a system that takes fingerprint ID and speech as input from the customer and produces the required deposit or withdrawal form at the counter. It is an auto-trigger system with proximity sensors to detect the presence of customers. The customers could either use the fingerprint module for auto-filling the account details or they can speak to the system in their own regional language. In the latter case, the system takes both the account and amount details, whereas in the former case, the customer needs to provide the amount details only. The entire process is done through voice communication between the system and the customer. The system uses speech recognition modules to recognize the customer requirements and performs them. The generated form is then directly sent to the counter and a token is provided to the customer according to which he/she can appear at the counter for further operations.

Keywords*: Artificial Intelligence, Voice Translation, Voice Recognition, Voice To Text, Text To Voice*

1. INTRODUCTION

In the current scenario of a bank, the deposit or withdrawal form filling is done manually by the customers. It is easier for an educated person to fill out the form but it is different in the case of an old-aged or illiterate person. They require the help of a second person in the bank in order to fill in their forms. To solve such a problem we are introducing an intelligent system. The system aims at generating a filled deposit or withdrawal form by taking speech as input. The output of the system is the filled deposit or withdrawal form generated as a print out from the system. The inputs given are the account number and the amount to be deposited or withdrawn, in the form of speech into the system.

The customer is directed to press a button that triggers the system. Then the system responds to the trigger by asking the customer to provide the required input details. The customer will then have to provide the input in the form of voice. After acquiring and verifying the input, the system is now ready to process the required form. The system then generates a filled form which will be printed automatically. The system consists of four modules, speech to text conversion module, text to speech conversion module, pdf generation module and an audio playing module. The speech to text conversion module converts the input given as a speech to text so that required operations can be performed by the system. The text to speech conversion module converts the questions that the system would like to ask the user, i.e in the form of text to speech. Pdf generation module helps in filling the deposit or withdrawal forms with the required data. Through audio play module, the system asks questions to get the required input from the user.

2. SYSTEM STUDY

System analysis is concerned with analyzing, designing, implementing and evaluating information system in our organization. It is carried out to make the system more effective by this modification or by the substantial redesign. In system analysis we identify the problem, study the alternate solution and select the most suitable solution, which meets the technical, economic and social demands for analysis, various tools such as data flow diagrams, interviews, on-site observation, etc., are used.

2.1 Existing System

In the present system, there is a fill out the form in the bank for both deposit and withdrawal. The form contains spaces to enter the account number and amount to be deposited or withdrawn. It is filled manually by the customers. In such cases, the form cannot be filled by the old-aged or illiterate people. So, they would require the help of a second person in the bank. Also, the customers can make mistakes while filling the form manually since at present the form contains smaller words which can cause confusion.

2.1.1 Demerits

- The old-aged and illiterate people require the help of a second person to fill in their forms.
- It is a time-consuming process.
- The forms contain smaller words that can cause confusion and produce errors while entering the account number and amount.

2.2 Proposed System
The proposed system is an automatic form filling and generating a system that can be used in a bank. This system has more advantages than the current system. This system produces the filled deposit or withdrawal forms based on the input provided by the customer's i.e. the account number and amount to deposit or withdraw as speech. Old-aged and illiterate people need not look for an educated person in the bank to their forms. This system takes speech input so it is easier to use.

2.2.1 Merits
- **No manual filling required**
  The customers need not fill the form manually. They will only need to input the account number and amount to deposit or withdraw in the form of speech.
- **Accurate**
  The input account number and amount will be entered into the form and the generated form will contain correct values for account number and amount as entered.
- **Fast result**
  The system is fast in generating the required deposit or withdrawal form as soon as the input is entered.
- **The help of a second person is not required**
  Old aged and illiterate people can fill their deposit or withdrawal forms without the help of a second person in the bank.
- **Easy to use**
  The system takes the input account number and amount to deposit or withdraws through speech i.e. the customer only needs to answer the questions asked by the system. So, it is very easy and can be used by anyone.

3. REQUIREMENT ANALYSIS
3.1 Hardware:
• Processor: Intel Pentium dual core
• RAM: 4GB
• Hard disk drive:40GB and above
• Speed: Above 1.5GHZ
• Microphone
• Speakers
• Printer
• Display monitor

3.2 Software:
• Operating System-Windows family
• language: Python
• IDE: Spyder

4. SYSTEM DESIGN
The purpose of the design is to plan the solution of a problem specified by the requirement document. The design of the system is the most critical factor affecting the quality of the software and has a major impact on the later phases, particularly testing and maintenance. This phase starts with the requirement document delivered by the maps the requirements into architecture. This architecture defines the components, their interfaces and behaviors. System design aims to identify the modules that should be in the system, the specifications of these modules and their interactions to produce the desired results. Since the system is a user interface, good user interface design is critical to the success of a system. The main constraints considered in the design are:

- Recoverability
- User familiarity
- Consistency

4.1 Module Description
The different modules are:
- Speech to Text conversion module
- Text to Speech conversion module
- PDF file generation module
- Audio playing module

4.1.1 Speech to Text Conversion Module
Speech to text conversion is the process of converting spoken words into written texts. This process is also often called speech recognition. Although these terms are almost synonymous, Speech recognition is sometimes used to describe the wider process of extracting meaning from speech, i.e. speech understanding. The term voice recognition should be avoided as it is often associated with the process of identifying a person from their voice. In this module, user can speak to the system. The system listens to the user and converts it into text. All speech-to-text systems rely on at least two models: an acoustic model and a language model. In addition, large vocabulary systems use a pronunciation model. It is important to understand that there is no such thing as a universal speech recognizer. To get the best transcription quality, all of these models can be specialized for a given language, dialect, application domain, type of speech, and communication channel. Like any other pattern recognition technology, speech recognition cannot be error free. The speech transcript accuracy is highly dependent on the speaker, the style of speech and the environmental conditions. Speech recognition is a harder process than what people commonly think, even for a human being. Humans are used to understanding speech, not to transcribing it, and only speech that is well formulated can be transcribed without ambiguity. From the user's point of view, a speech-to-text system can be categorized based in its use: command and control, dialog system, text dictation, audio document transcription, etc. Each user has specific requirements in terms of latency, memory constraints, vocabulary size, and adaptive features.

4.1.2 Text To Speech Conversion Module
A text-to-speech system (or "engine") is composed of two parts: a front-end and a back-end. The front-end has two major tasks. First, it converts raw text containing symbols like numbers and abbreviations into the equivalent of written-out words. This process is often called text normalization, pre-processing, or tokenization. The front-end then assigns phonetic transcriptions to each word and divides and marks the text into prosodic units, like phrases, clauses, and sentences. The process of assigning phonetic transcriptions to words is called text-to-phoneme or grapheme-to-phoneme conversion. Phonetic transcriptions and prosody information together make up the symbolic linguistic representation that is output by the front-end. The back-end-often referred to as the synthesizer-then converts the symbolic linguistic representation into sound. In certain systems, this part includes the computation of the target prosody (pitch contour, phoneme durations), which is then imposed on the output speech System converts the text into audio and saves as an mp3 file user can give text. A text-to-speech (TTS) system converts normal language text into speech; other systems render symbolic linguistic representations like phonetic transcriptions into speech. The quality of a speech synthesizer is judged by its similarity to the human voice and by its ability to be understood clearly. An intelligible text-to-speech program allows people with visual impairments or reading disabilities to listen to written words on a home computer.

4.1.3 PDF Generation Module
Programmatically creating PDFs is fairly common among server-side and desktop applications. The PDF format is a document

presentation format. HTML which probably more readers of this blog are familiar with is also a sort of document presentation format but the two have different goals. PDF is designed for fixed layouts on specific sizes of paper while HTML uses a variable layout that can differ across screen sizes, screen types and potentially across browsers. With relatively few exceptions, the same document should look exactly the same on two different PDF viewers even on different operating systems, screen sizes or form factors. The pdf generation module is used to generate the deposit or withdrawal form as per the user requirement. The generated form contains the account number and the amount to be deposited or withdrawn. The pdf generation module writes the input information i.e. the account number and the amount on a page that is saved in the form of a pdf file. To generate a pdf in python, report lab module is used. The statements required to generate a pdf using reportlab module is given below:

```
from reportlab.pdfgen import canvas         //Statement 1
c=canvas.Canvas("Output.pdf")               //Statement 2
c.drawImage('SBI-logo.png', 100, 770,       //Statement 3
mask='auto', width=50, height=50)
c.setFont("Helvetica", 30)                  //Statement 4
c.drawString(190,780,"STATE BANK            //Statement 5
OF INDIA")
```

Statement 1 imports the module reportlab.pdfgen to perform the required operations on a pdf.
Statement 2 generates a pdf file called 'Output.pdf' with a single page.
Statement 3 contains a function called draw Image that is used to insert an image into the generated pdf file. The draw Image function takes different attributes of the image as its parameter, like the width attribute shown in the above code statement.
Statement 4 contains a function called set Font that is used to set a font type to the pdf. Different types of fonts are available for this function.
Statement 5 contains a function called drawString that is used to write a string on the pdf file.

4.2 Auto Audio playing module

The module used for automatically playing audio in this project is the Pygame module of python. This module contains classes for loading Sound objects and controlling playback.

Pygame module for loading and playing sounds contains the following:

pygame.mixer.init: initialize the mixer module.
pygame.mixer.pre_init: preset the mixer init arguments
pygame.mixer.quit: uninitialize the mixer
pygame.mixer.get_init: test if the mixer is initialized
pygame.mixer.stop: stop playback of all sound channels
pygame.mixer.pause: temporarily stop playback of all sound channels
pygame.mix-er.unpause: resume paused playback of sound channels
pygame.mixer.fadeout: fade out the volume on all sounds before stopping
pygame.mixer.set_num_channels: set the total number of playback channels
pygame.mixer.get_num_channels: get the total number of playback channels
pygame.mixer.set_reserved: reserve channels from being automatically used
pygame.mixer.find_channel: find an unused channel
pygame.mixer.get_busy: test if any sound is being mixed
pygame.mixer.Sound: Create a new Sound object from a file or buffer object
pygame.mixer.Channel: Create a Channel object for controlling playback.

The mixer module is optional and depends on SDL_mixer. Your program should test that pygame.mixer is available and initialized before using it. The mixer module has a limited number of channels for playback of sounds.

Usually, programs tell pygame to start playing audio and it selects an available channel automatically. The default is 8 simultaneous channels, but complex programs can get more precise control over the number of channels and their use. All sound playback is mixed in background threads. When you begin to play a Sound object, it will return immediately while the sound continues to play. A single Sound object can also be actively played back multiple times. The mixer also has a special streaming channel. This is for music playback and is accessed through the module. The mixer module must be initialized like other pygame modules, but it has some extra conditions. The pygame.mixer.init() function takes several optional arguments to control the playback rate and sample size. Pygame will default to reasonable values, but pygame cannot perform Sound resampling, so the mixer should be initialized to match the values of your audio resources. The default is set to reduce the chance of scratchy sounds on some computers. You can change the default buffer by calling pygame.mixer.pre_init() before pygame.mixer.init() is called. pygame.mixer.Sound contains the following :

pygame.mixer.Sound.play: begin sound playback
pygame.mixer.Sound.stop: stop sound playback
pygame.mixer.Sound.fadeout: stop sound playback after fading out
pygame.mixer.Sound.set_volume: set the playback volume for this Sound
pygame.mixer.Sound.get_volume: get the playback volume
pygame.mixer.Sound.get_num_channels: count how many times this Sound is playing
pygame.mixer.Sound.get_length: get the length of the Sound
pygame.mixer.Sound.get_raw: return a bytestring copy of the Sound samples.

5. IMPLEMENTATION

Implementation literally means to put into effect what to carry out. The system implementation phase of the software deals with the translation of the design specification into the source code. The ultimate goal of the implementation is to write the source code and internal documentation. It is the process of computer programming, documenting, testing, and bug fixing involved in creating and maintaining applications and frameworks resulting in a software product. The system flow starts simple running on packages, sample output etc. is a part of implementation.
By implementing the project we have taken into consideration to meet the goals:

- Clarity and simplicity of code.
- Accuracy of results.
- Minimization of response code.
- Minimization of amount of memory used.

5.1 Speech to text conversion module

Speech recognition is the process of converting spoken words to text. Python supports many speech recognition engines and APIs, including Google Speech Engine, Google Cloud Speech API, Microsoft Bing Voice Recognition and IBM Speech to Text. In our project, we are using Google Speech Recognition Engine with

Python. Speech to text is converted using the Speech Recognition library which uses the Microphone as the source for obtaining the speech. listen() function listens to the microphone source and recognize_google() function converts the obtained speech to text.

```
import speech_recognition as sr
r = sr.Recognizer()
with sr.Microphone() as source:
audio = r.listen(source)
text = r.recognize_google(audio)
```

5.2 Text to speech conversion module
There are several APIs available to convert text to speech in python. One such APIs is the Google Text to Speech API commonly known as the gTTS API. gTTS is a very easy to use tool which converts the text entered, into audio which can be saved as an mp3 file.

```
from gtts import gTTS
import os
tts = gTTS(text='provide the text here', lang='en', slow= True)
tts.save("audio.mp3")
os.system("audio.mp3")
```

5.3 Pdf file generation module
The Portable Document Format (PDF) lets you create documents that look exactly the same on every platform. Our project requires the dynamic creation of PDF document at the end of user interaction. This is done by using the reportlab.pdfgen library.

```
from reportlab.pdfgen import canvas
c=canvas.Canvas("Output.pdf")
c.drawImage('SBI-logo.png', 100, 770, mask='auto', width=50,
height=50)
c.setFont("Helvetica", 30)
c.drawString(190,780,"STATE BANK OF INDIA")
```

5.4 Audio playing module
In this module, the pre-saved audios are made to play at the required time that suits the situation of the system's working. This is done by importing the pygame module in code. The function pygame.init() initializes all the imported pygame modules. The function pygame.mixer.music.load() loads the given audio and pygame.mixer.music.play() plays the audio is loaded.

```
import pygame
pygame.init()
pygame.mixer.music.load("audio.mp3")
pygame.mixer.music.play()
```

6. TESTING
The aim of the software testing is to identify the defects in the program. However, in practice even after satisfactory completion of the testing phase, it is not possible to guarantee that a program is error free. This is because the input domain of most programs is very large, and it is not practical to test the program exhaustively with respect to each value that the input can assume. Even with this obvious limitation of testing process, we should not underestimate the importance of testing. We must remember that careful testing can expose most of the defects existing in the program, and therefore provides a practical way of reduce defects in the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved.

6.1 Unit testing
Unit testing is carried out to screen wise, each screen being identified as an object. Attention is diverted to individual modules, independently to one another to locate the error in coding and logic. In unit testing:

- Module interface is tested to ensure that information properly flows into and out of the units under test.
- Boundary conditions are tested to ensure that the modules operate properly at boundaries established to limit or restrict processing.
- All independent paths through the control structures are executed to ensure all statements in the module have executed at least once.
- The local data structure is tested to ensure that the data stored temporarily maintains its integrity during execution.
- Exception handling paths are also tested.

Table 6.1: Deposit or Withdraw

Sample input (voice)	Expected output	Obtained Output
Deposit/Withdraw/Withdrawal	Detected Successfully	Detected Successfully
Invalid cases	Not detected, ask for Deposit or Withdraw	Not detected, ask for Deposit or Withdraw

Table 6.2: Account Number

Sample input (voice)	Expected output	Obtained output
Valid account number	Detected Successfully	Detected Successfully
Invalid account number	Not detected, ask for a valid account number	Not detected, ask for a valid account number

Table 6.3: Amount

Sample input (voice)	Expected output	Obtained output
Valid account number	Detected successfully	Detected successfully
Invalid account number	Not detected, ask for the valid amount	Not detected, ask for the valid amount

6.2 Integration Testing
Integration testing is the systematic way of constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. Unit tested modules were taken and a single program structure is built as per the system design. Incremental integration is adopted here. The entire software was developed and tested in small segments, where errors are easy to locate rectify. Each database manipulation operations are written as single java files and separately tested and combined to form a single program. After integration, the single program was tested again with numerous test data to check for its functionality. The integration can be performed in two ways:

- Top-down integration.
- Bottom-up integration.
-

Here bottom-up integration is performed and is implemented with following steps:

- Lower level modules are combined to form clusters that perform a specific software subfunction.
- Then the clusters are tested.

6.3 Validation Testing

Validation checks are performed on the following fields. Text field: The text fields can contain only the number of characters less than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes an error message. Numeric field: The numeric field can contain only numeric values. An entry of character flashes an error message.

Table 6.4: Deposit or Withdraw

Field	Sample Input (voice)	Expected output	Obtained output
Deposit/ Withdraw	Deposit	Detected Successfully	Detected Successfully
Deposit/ Withdraw	Withdraw	Detected Successfully	Detected Successfully
Deposit/ Withdraw	Withdraw	Detected Successfully	Detected Successfully
Deposit/ Withdraw	Hello	Not detected, Try again	Not detected, Try again
Deposit/ Withdraw	abcde	Not detected, Try again	Not detected, Try again

Table 6.5: Account Number

Field	Sample input (voice)	Expected output	Obtained output
Account Number	121201352	Detected Successfully	Detected Successfully
Account Number	2222222222	Detected Successfully	Detected Successfully
Account Number	451735	Not detected, Try again with valid account number	Not detected, Try again with valid account number
Account Number	abcd5wbm	Not detected, Try again with valid account number	Not detected, Try again with valid account number
Account Number	3456213td	Not detected, Try again with valid account number	Not detected, Try again with valid account number

Table 6.6: Amount

Field	Sample input (voice)	Expected output	Obtained output
Amount	Thousand one hundred and fifty	1500	1150
Amount	Nine thousand nine hundred and ninety nine	9999	9999
Amount	One lakh	100000	One lakh
Amount	Five million	5000000	Five million
Amount	tdwed634	Invalid amount, Try again with valid amount	Invalid amount, Try again with valid amount

7. CONCLUSION AND FUTURE SCOPE

In the recent past we have seen a great change in the banking sectors, that is the deposit and withdrawal which was till then a difficult process has become much easier with the implementation of ATM machines thereby helping the common men to a greater extend. But the withdrawal and deposit form filling is still difficult. For overcoming this difficulty GIF has been designed, implemented and tested successfully in this project. This software provides a filled out deposit or withdrawal form just by taking speech as input. Thus it makes both the common men and the old aged to easily fill the forms. The project can be extended to regional languages and can even be linked with the bank databases and the form can be directly sent to the officials.

Thus printed deposit and withdrawal forms will be no longer needed. Similarly proper queuing system can be implemented so that people will not have to wait in the queue. By using the fingerprint sensors users account can be automatically linked, thus users will not have to enter their account number. The project meets the entire requirement as described in the proposed system and satisfies user requirements to a great extent. The software thus developed is easy to maintain and is quite user-friendly.

8. ACKNOWLEDGMENT

At the very outset, we would like to give the first honors to God who gave us the knowledge and wisdom to complete this main project.

This main project would not have been a success without the inspiration, valuable suggestion and moral support from many great people to whom we owe our sincere gratitude.

We would like to express our sincere gratitude to our college, Vimal Jyothi Engineering College that gave us an opportunity to do this mini project. We are grateful to the principal Dr. Benny Joseph who is the leading light of our institution.

We express our deepest sense of gratitude to Ms. Ancy K Sunny, our main project guide, for her valuable support, advice and involvement, which was a major source of inspiration and encouragement.

We are thankful to Ms. Akhila Mathew, who coordinated the project work in the best possible way. We express our sincere gratitude to Dr. Manoj V Thomas, the honorable Head of the Department of Computer Science and Engineering for his constant guidance and support throughout the main project.

We wish to extend our sincere thanks to all the Assistant Professors in the department of Computer Science and Engineering for their valuable support and guidance.

We bear in hear every gestures of care, act of kindness and every bit of friendship extended to us by all the people, who helped as in different ways during the work of our main project. We truly admire our parents and friends for their constant encouragement and enduring support which is inevitable for the success of our venture.

9. REFERENCES

[1] http://pythoncoder7.blogspot.in/2014/06/pyqt-rapid-gui-programming-with-python.html

[2] http://typea.info/blg/glob/2016/03/python-gui-spyder-qt-designer-gui.html

[3] https://pythonspot.com/speech-recognition-using-google-speech-api/

[4] http://code.activestate.com/recipes/579115-recognizing-speech-speech-to-text-with-thepython-/

[5] https://gist.github.com/Sharon-f/cf59edf3a27616db5a91bc27b2237e81

[6] https://gist.github.com/ericollargiu/cd6e10ca9b0514f61d0e0f2e45ccc9c4
[7] http://www.include.gr/debian/mpg321/
[8] https://docs.python.org/3/library/tkinter.html
[9] https://github.com/baruchel/txt2pdf
[10] https://pythonprogramminglanguage.com/text-to-speech/
[11] http://effbot.org/tkinterbook/button.htm
[12] https://www.youtube.com/watch?v=30oWH6yKuB4
[13] https://ubuntuforums.org/showthread.php?t=2371635
[14] https://stackoverflow.com/questions/48243522/python-how-do-i-get-my-gui-todisplay-in-spyder
[15] https://doc.qt.io/archives/4.3/designer-getting-started.html
[16] https://www.pythoncentral.io/category/python-library-tutorials/
[17] http://hetland.org/writing/instant-python.html
[18] https://stackoverflow.com/questions/2252726/how-to-create-pdf-files-in-python

BIOGRAPHIES

Aiswarya ManojKumar
Student
Computer Science and Engineering

Anagha Pallen Pavithran
Student
Computer Science and Engineering

Bagya Girish
Student
Computer Science and Engineering

Sneha C K
Student
Computer Science and Engineering