**Project Report: Voice-to-Text Form Automation for Banking**

**Project Overview**

This project integrates voice recognition and a graphical user interface (GUI) to create a digital banking form that enhances user interaction by allowing voice-to-text input. Built with Python, the system combines speech recognition and tkinter libraries to provide a seamless experience for entering data with minimal typing, streamlining the data entry process in the banking sector.

**Objectives**

1. Develop a banking form GUI for collecting basic information, including:
   * Full Name
   * Account Number
   * Phone Number
   * Address
   * Aadhaar Card Number
   * Withdrawal Amount
2. Incorporate a voice-to-text feature that captures audio input and translates it into text for each form field.
3. Ensure a user-friendly interface that allows data entry with voice commands and displays submission confirmation.

**Technology Stack**

* **Python:** Programming language used for building the application.
* **Tkinter:** GUI library used to create the form interface.
* **Speech Recognition Library:** Provides the speech-to-text functionality, utilizing Google’s API for audio transcription.

**Code Walkthrough**

**1. Voice-to-Text Conversion**

The voice\_to\_text function uses the speech\_recognition library:

* **Recognizer:** Initializes an instance for handling audio recognition.
* **Microphone as Source:** Uses a microphone as the input device for capturing audio.
* **Google Speech Recognition:** Transcribes audio to text.
* **Exception Handling:** Provides warnings and errors if the voice input is unintelligible or if there is an issue with the API.

**2. GUI Design with Tkinter**

* **Form Layout:** Each field in the form includes a label, an entry box, and a button for voice input.
* **Voice Input Button:** For each entry field, a microphone button allows users to dictate input.
* **Submit Button:** Submits the form data, capturing and displaying the entered details in a message box.

**3. Methods in Banking Form App**

* **create\_field:** Initializes labels, entry fields, and voice input buttons.
* **fill\_with\_voice:** Uses the voice\_to\_text function to insert recognized text into the respective entry field.
* **submit\_form:** Collects all form data upon submission and displays it in a message box, simulating form submission.

**Project Flowchart**

Here’s a simple text-based flowchart layout for the Voice-to-Text Banking Form application:

+-------------------+

| User |

+-------------------+

|

v

+---------------------------+

| Open Banking Form App |

+---------------------------+

|

+---------------------------+

| Voice-to-Text |

| Module |

+---------------------------+

|

|

+---------------------------+

| Capture Voice Input |

+---------------------------+

|

v

+---------------------------+

| Convert Audio to Text |

+---------------------------+

|

|

+---------------------------+

| Error Handling |

| (if voice input fails) |

+---------------------------+

|

v

+---------------------------+

| Display Form |

| Fields and Enter |

| Text |

+---------------------------+

|

v

+---------------------------+

| Submit Form |

+---------------------------+

|

v

+---------------------------+

| Display Submission |

| Confirmation |

+---------------------------+

This ASCII-style layout represents each process step and shows the logical flow. It should look fine on Notepad, though using a fixed-width font (like Consolas or Courier New) will help keep it aligned.

**Implementation Code**

import speech\_recognition as sr

import tkinter as tk

from tkinter import messagebox

# Function to capture voice input and convert it to text

def voice\_to\_text():

recognizer = sr.Recognizer()

with sr.Microphone() as source:

print("Listening for voice input...")

audio = recognizer.listen(source)

try:

text = recognizer.recognize\_google(audio)

print(f"Recognized Text: {text}")

return text

except sr.UnknownValueError:

print("Could not understand the audio")

messagebox.showwarning("Warning", "Could not understand the audio. Please try again.")

except sr.RequestError:

print("Error with the API")

messagebox.showerror("Error", "There was an issue with the Speech Recognition API. Check your connection.")

# GUI form for banking sector data

class BankingFormApp:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Banking Sector Form")

# Creating form labels and fields

self.create\_field("Full Name", 0)

self.create\_field("Account Number", 1)

self.create\_field("Phone Number", 2)

self.create\_field("Address", 3)

self.create\_field("Adhaar Card", 4)

self.create\_field("Withdraw Amount", 5)

# Submit button

submit\_button = tk.Button(root, text="Submit", command=self.submit\_form)

submit\_button.grid(row=6, column=1, pady=10)

def create\_field(self, label\_text, row):

label = tk.Label(self.root, text=label\_text)

label.grid(row=row, column=0, padx=10, pady=5, sticky="w")

entry = tk.Entry(self.root, width=40)

entry.grid(row=row, column=1, padx=10, pady=5)

# Voice Input button

voice\_button = tk.Button(self.root, text="🎙️", command=lambda e=entry: self.fill\_with\_voice(e))

voice\_button.grid(row=row, column=2, padx=5, pady=5)

def fill\_with\_voice(self, entry):

text = voice\_to\_text()

if text:

entry.delete(0, tk.END)

entry.insert(0, text)

def submit\_form(self):

# Get all entries and show as a message box (can replace with form submission functionality)

info = "\n".join(f"{self.root.grid\_slaves(row=i, column=0)[0]['text']}: {self.root.grid\_slaves(row=i, column=1)[0].get()}" for i in range(6))

messagebox.showinfo("Form Submitted", f"Details:\n{info}")

# Run the application

if \_\_name\_\_ == "\_\_main\_\_":

root = tk.Tk()

app = BankingFormApp(root)

root.mainloop()

A screenshot of a computer

Description automatically generated

**Testing and Evaluation**

* **Voice Accuracy:** Testing included speaking varied names, account numbers, and other details to evaluate accuracy.
* **Error Handling:** Verified proper warnings and error messages for unknown audio input and API errors.
* **Usability:** Users could fill forms without typing, making it suitable for people with physical limitations or for increasing efficiency in high-traffic banking environments.

**Future Enhancements**

1. **Database Integration:** Add database connectivity for storing submitted form data.
2. **Multi-Language Support:** Enable multilingual voice recognition to broaden usability.
3. **Enhanced Error Feedback:** Provide more detailed API error messages for troubleshooting.

**Key Use Cases Of This Application**

**1. Banking Customer Service Centres**

**Use Case**: Streamlining Data Entry for Customer Representatives

* **Description**: In customer service centers, agents frequently fill out forms for client requests such as account opening, updates, loan applications, and more. This solution allows agents to fill forms quickly using voice commands, reducing typing time and minimizing manual errors.
* **Benefits**: Improves processing speed and allows agents to focus on customer interaction rather than data entry.

**2. In-Branch Services for Enhanced Accessibility**

**Use Case**: Assisting Customers with Disabilities

* **Description**: Many banking customers face challenges in completing forms due to visual impairments, physical disabilities, or age-related issues. By providing a voice-to-text interface, banks can enable these customers to complete forms with ease and independence.
* **Benefits**: Enhances accessibility for differently-abled and elderly customers, supporting inclusivity in banking services.

**3. Remote Banking Assistance**

**Use Case**: Assisting Customers with Form-Filling Over the Phone

* **Description**: Banks often provide remote assistance for filling out forms, especially for online services like loan applications or credit card requests. Voice-to-text can facilitate this process by allowing customers to dictate form responses over the phone, which the system transcribes and saves.
* **Benefits**: Increases convenience for remote customers and reduces dependency on physical branches.

**4. Financial Advisors and Mobile Banking Agents**

**Use Case**: Assisting Financial Advisors During Client Meetings

* **Description**: Financial advisors often work on the go and need to record client information quickly and efficiently. This tool can help them fill out forms through voice input while interacting with clients, which they can later upload or process in the bank’s system.
* **Benefits**: Increases productivity for mobile agents by enabling hands-free data entry, allowing them to focus more on client interaction.

**5. Automated Data Collection for Loan and Credit Applications**

**Use Case**: Simplifying Loan or Credit Application Forms

* **Description**: Loan or credit application forms are often lengthy, requiring detailed information from applicants. This voice-to-text solution can streamline the application process, enabling applicants to fill out sections verbally.
* **Benefits**: Reduces the complexity of filling out large forms and ensures that data collection is quick and efficient, potentially improving loan processing times.

**6. KYC (Know Your Customer) and Compliance Updates**

**Use Case**: Simplifying KYC and Compliance Information Collection

* **Description**: Banks frequently collect KYC information as part of regulatory compliance, which can be repetitive and time-consuming. By integrating voice-to-text for KYC data updates, banks can gather client information more effectively while maintaining data accuracy.
* **Benefits**: Accelerates the KYC data collection process, reducing the workload for both customers and bank employees while ensuring accurate data capture.

**7. Support for Language Flexibility in Multilingual Regions**

**Use Case**: Providing Multi-Language Form Assistance

* **Description**: In regions with diverse languages, this voice-to-text tool can be customized to recognize multiple languages, allowing customers to interact with the form in their preferred language. It can help bridge communication gaps in multilingual areas.
* **Benefits**: Enhances user experience and inclusivity, allowing customers to feel more comfortable interacting with banking services in their native language.

**8. Data Collection in Non-Traditional Banking Areas**

**Use Case**: Assisting Field Agents in Collecting Data in Rural Areas

* **Description**: In rural or remote areas where literacy or digital literacy may be lower, field agents can use this tool to help customers provide the necessary details for account openings or government-subsidized banking schemes, which the system then processes and stores.
* **Benefits**: Supports financial inclusion by enabling underserved populations to access banking services more easily.

**9. Corporate Banking and Internal Staff Use**

**Use Case**: Internal Form-Filling for Staff

* **Description**: Corporate banking or internal banking processes, such as updating employee records, managing compliance forms, and processing HR requests, require regular form completion. This tool can assist employees by enabling quick, hands-free form entry.
* **Benefits**: Enhances efficiency for internal operations and reduces manual data entry time for employees.

**10. Documenting Customer Complaints and Feedback**

**Use Case**: Voice-to-Text for Customer Feedback Collection

* **Description**: Collecting feedback or complaints via voice input is useful for quick documentation, particularly for long and detailed responses. The tool can transcribe customers’ responses directly into feedback forms for future review and action.
* **Benefits**: Improves data capture speed and accuracy for feedback collection, making it easier to act on customer concerns.

**Conclusion**

This project successfully demonstrated how voice recognition can streamline data entry in the banking sector. The system is flexible for further improvements, offering a base for applications where voice-enabled form filling enhances accessibility and efficiency.

**My Project Link**

https://github.com/Sahil4000/SpeechReco.git