In [1]:
```
pip install captcha
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: captcha in c:\users\yash dhumal\appdata\roaming\pyt
hon\python311\site-packages (0.6.0)
Requirement already satisfied: Pillow in d:\programs\lib\site-packages (from captc
ha) (9.4.0)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
pip install pyttsx3
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pyttsx3 in c:\users\yash dhumal\appdata\roaming\pyt
hon\python311\site-packages (2.90)
Requirement already satisfied: comtypes in c:\users\yash dhumal\appdata\roaming\py
thon\python311\site-packages (from pyttsx3) (1.4.5)
Requirement already satisfied: pypiwin32 in c:\users\yash dhumal\appdata\roaming\p
ython\python311\site-packages (from pyttsx3) (223)
Requirement already satisfied: pywin32 in d:\programs\lib\site-packages (from pytt
sx3) (305.1)
Note: you may need to restart the kernel to use updated packages.
```

In [5]:
```
pip install Pillow
```

```
Defaulting to user installation because normal site-packages is not writeableNote:
you may need to restart the kernel to use updated packages.

Requirement already satisfied: Pillow in d:\programs\lib\site-packages (9.4.0)
```

In [6]:
```
import tkinter as tk
from tkinter import messagebox
from captcha.image import ImageCaptcha
import pyttsx3
import random
import string
import os

class CaptchaGeneratorApp:
    def __init__(self, root):
        self.root = root
        self.root.title("CAPTCHA Generator and Verifier")

        # GUI elements
        self.label_length = tk.Label(self.root, text="Enter CAPTCHA Length:")
        self.label_length.pack(pady=10)

        self.entry_length = tk.Entry(self.root)
        self.entry_length.pack()

        self.button_generate = tk.Button(self.root, text="Generate CAPTCHA", commar
        self.button_generate.pack(pady=10)

        self.label_captcha_image = tk.Label(self.root, text="CAPTCHA Image:")
        self.label_captcha_image.pack()

        self.label_captcha_audio = tk.Label(self.root, text="CAPTCHA Audio:")
        self.label_captcha_audio.pack()

        self.label_user_input_image = tk.Label(self.root, text="Enter Image CAPTCH
        self.label_user_input_image.pack()

        self.entry_user_input_image = tk.Entry(self.root)
        self.entry_user_input_image.pack()
        self.label_user_input_audio = tk.Label(self.root, text="Enter Audio CAPTCHA
        self.label_user_input_audio.pack()

        self.entry_user_input_audio = tk.Entry(self.root)
        self.entry_user_input_audio.pack()

        self.button_verify = tk.Button(self.root, text="Verify CAPTCHA", command=se
        self.button_verify.pack(pady=10)

        # Initialize CAPTCHA variables
        self.captcha_text = ""
        self.captcha_image_file = ""
        self.captcha_audio_file = ""

    def generate_captcha(self):
        try:
            length = int(self.entry_length.get())
            if length <= 0:
                messagebox.showerror("Error", "Length must be a positive integer.")
                return

            # Generate CAPTCHA text
            self.captcha_text = generate_captcha_text(length)

            # Generate CAPTCHA image
            self.captcha_image_file = generate_image_captcha(self.captcha_text)
            self.label_captcha_image.config(text=f"CAPTCHA Image: {self.captcha_ima

            # Generate CAPTCHA audio
            self.captcha_audio_file = generate_audio_captcha(self.captcha_text)
            self.label_captcha_audio.config(text=f"CAPTCHA Audio: {self.captcha_auc

            # Play audio CAPTCHA
            self.play_audio_captcha(self.captcha_audio_file)

        except ValueError:
            messagebox.showerror("Error", "Invalid input. Please enter a valid inte
        except Exception as e:
            messagebox.showerror("Error", f"Unexpected error: {e}")

    def play_audio_captcha(self, audio_file):
        # Play the audio CAPTCHA using the system's default audio player
        try:
            if os.name == 'nt':  # For Windows
                os.system(f'start {audio_file}')
            elif os.name == 'posix':  # For Linux and MacOS
                os.system(f'afplay {audio_file}')  # MacOS
                # os.system(f'aplay {audio_file}')  # Linux
        except Exception as e:
            messagebox.showerror("Error", f"Unable to play audio: {e}")

    def verify_captcha(self):
        if not self.captcha_text:
            messagebox.showerror("Error", "Generate CAPTCHA first.")
            return

        user_input_image = self.entry_user_input_image.get()
        user_input_audio = self.entry_user_input_audio.get()

        # Verify CAPTCHA
        if verify_captcha(user_input_image, self.captcha_text) and verify_captcha(u
            messagebox.showinfo("Success", "CAPTCHA verification successful!")
        else:
```

```python
            messagebox.showerror("Error", "CAPTCHA verification failed.")

def generate_captcha_text(length):
    characters = string.ascii_letters + string.digits
    captcha_text = ''.join(random.choice(characters) for _ in range(length))
    return captcha_text

def generate_image_captcha(text):
    image = ImageCaptcha(width=280, height=90, fonts=None, font_sizes=None)
    captcha = image.generate(text)
    image_file = f'captcha_{text}.png'  # Save the CAPTCHA image to file
    image.write(text, image_file)

    # Load the generated image and convert it to black and white
    from PIL import Image
    img = Image.open(image_file)
    bw_img = img.convert('L')  # Convert to grayscale
    bw_img.save(image_file)

    return image_file

def generate_audio_captcha(text):
    engine = pyttsx3.init()
    audio_file = f'captcha_{text}.mp3'
    try:
        engine.save_to_file(text, audio_file)
        engine.runAndWait()
        print(f"Audio file {audio_file} generated successfully")
    except Exception as e:
        print(f"Failed to generate audio: {e}")
    return audio_file

def verify_captcha(input_text, captcha_text):
    return input_text.lower() == captcha_text.lower()

if __name__ == "__main__":
    root = tk.Tk()
    app = CaptchaGeneratorApp(root)
    root.mainloop()
```

```
Audio file captcha_429r.mp3 generated successfully
```

In [ ]: