#VIRTUNEXA WEEK 1 TASK :

GRAPHICAL USER INTERFACE

Overview:

This script implements a countdown timer using a **Graphical User Interface (GUI)** built with the tkinter library. It features an interactive and visually appealing timer with start, pause/resume, and exit functionalities. The timer supports arithmetic expressions, time units (s for seconds, m for minutes), and real-time audio feedback using text-to-speech.

Dependencies:

The following Python libraries are required:

1. **tkinter**: For building the GUI.
2. **Pillow (PIL)**: To handle image resizing for the background.
3. **pyttsx3**: For text-to-speech audio output.
4. **re**: For validating and parsing the time input.

Install any missing dependencies using:

pip install pillow pyttsx3

Features

1. **Graphical User Interface**:
    - Customizable background image.
    - Input field for time with multiple formats.
    - Buttons for starting, pausing/resuming, and exiting the timer.
    - Real-time countdown display.
2. **Input Flexibility**:
    - Supports various formats like 30s, 5m, 2*30+15.
    - Allows arithmetic expressions and time unit conversions.
3. **Real-Time Feedback**:
    - Announces the remaining seconds during the last 5 seconds.
    - Announces "Time's up!" upon completion.
4. **Pause/Resume Functionality**:

  - Users can pause and resume the countdown seamlessly.

Components

1. CountdownTimer Class

This is the primary class that manages the GUI and timer logic.

Attributes

- **root**: The main tkinter window.
- **countdown_active**: Boolean flag to indicate whether the timer is active.
- **is_paused**: Boolean flag to track if the timer is paused.
- **time_left**: Remaining time in seconds.
- **countdown_job**: Reference to the after() job for the countdown loop.
- **engine**: Text-to-speech engine instance.

Methods

1. **__init__(root)**
   - Initializes the GUI, variables, and layout.
   - Configures window dimensions to match the screen resolution.
2. **setup_background()**
   - Loads and resizes a background image (background.jpg) to fit the window.
   - Displays the background image using a Label.
3. **create_widgets()**
   - Creates and places the GUI components:
     - **Entry widget** for time input.
     - **Buttons** for Start, Pause, and Exit.
     - **Time display label** to show the countdown.
4. **safe_eval(expr)**
   - Parses and evaluates time expressions safely.
   - Converts time units:
     - 30s becomes 30.
     - 5m becomes 5*60.
   - Validates the expression using regular expressions.
   - Raises ValueError for invalid inputs.

5. **start_countdown(event=None)**
    - Starts the countdown based on user input.
    - Disables the Start button and enables the Pause button.
    - Validates and converts the input to seconds using safe_eval().
6. **pause_resume_countdown()**
    - Pauses or resumes the countdown:
        - When paused: Stops the countdown loop.
        - When resumed: Restarts the countdown from where it left off.
7. **countdown()**
    - Core logic for the countdown loop.
    - Updates the time display every second.
    - Announces remaining seconds if the time is <= 5 seconds.
    - Ends the countdown when time runs out, announcing "Time's up!"
8. **reset_timer()**
    - Resets the timer to its initial state.
    - Re-enables the Start button and disables the Pause button.
9. **exit_app()**
    - Stops the text-to-speech engine.
    - Closes the application.

---

2. Main Function

The entry point of the script.

- **main()**:
    - Creates the tkinter root window.
    - Initializes and runs the CountdownTimer application.

Usage

1. **Running the Script** Run the script in the terminal:

```
python countdown_gui.py
```

2. **Interacting with the Timer**
    - Enter the desired time in the input field. Examples:

        - 30s for 30 seconds.
        - 5m for 5 minutes.
        - 2*30+15 for 75 seconds.
    - Press **Enter** or click **Start Countdown** to begin.
    - Use the **Pause** button to pause/resume the countdown.
    - Use the **Exit** button to close the application.

Error Handling

1. **Invalid Input**:
    - If the input format is invalid, an error messagebox appears.
    - Examples of invalid inputs:
        - abc (non-numeric).
        - -5s (negative time).
2. **Edge Cases**:
    - Input of 0 or less raises an error.