



Python Project

# Quiz Game

# MODULES USED

TKINTER

- This module is used to create Graphic User Interface ( GUI ) for making project attractive .

WIN32COM  
CLIENT

- This allow python program to automate tasks and interact with various Windows application .

PYGAME

- This module is used for handling graphics , inputs , game-loop and sound .

TIME

- Time module provides various time related functions , allowing us to work with time related tasks

RANDOM

- This module is used to generate random result from an non-empty sequence .

# KEY FEATURES :

## 1. Text-to-Speech (TTS) Functionality:

- The win32com.client is used to instantiate the SAPI.SpVoice object, allowing the program to **speak text out loud**. The speak\_text() function encapsulates the speaker.speak() method for convenience.



## 2. Audio Playback:

- The game plays an applause sound using the pygame library. The play\_audio() function initializes the pygame.mixer, loads an audio file (applause-2.mp3), and plays it.



## 3. Color Class (For Console Output):

- The Color class defines various ANSI color codes (for styling text output in the console, not in the GUI). For example, if the user loses, the text "You lost" will be printed in red.



## 4. Main Game Logic:

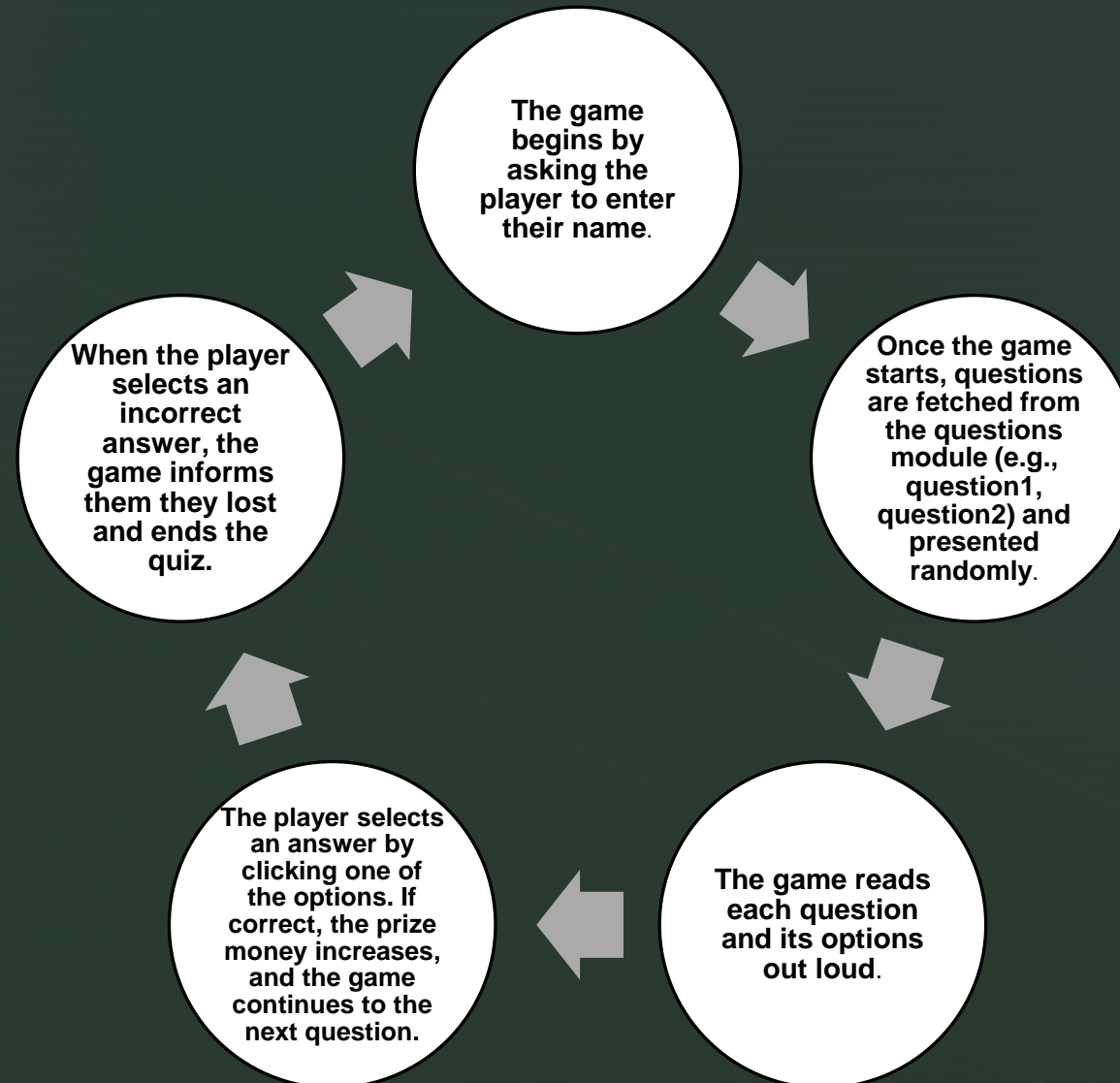
- The game maintains a game\_state dictionary to track the current question index, the money amounts for each question, and the player's prize.
- Questions are selected and displayed using display\_question(), which also calls speak\_text() to read the question and options aloud.
- The player selects an answer by clicking one of the buttons generated by tkinter. Their selection is validated by check\_answer(). If correct, they proceed to the next question; if incorrect, the game ends.



## 5. GUI Components:

- The main window is created using tkinter (root = tk.Tk()).
- A series of labels and buttons allow the player to enter their name, view questions, select answers, and see the results.
- The interface is dynamic, with labels and buttons updated based on the player's actions and game progression.
- The start\_button triggers the start\_quiz() function, which initializes the quiz by randomizing the question order and starting the first question.

# Game Flow:

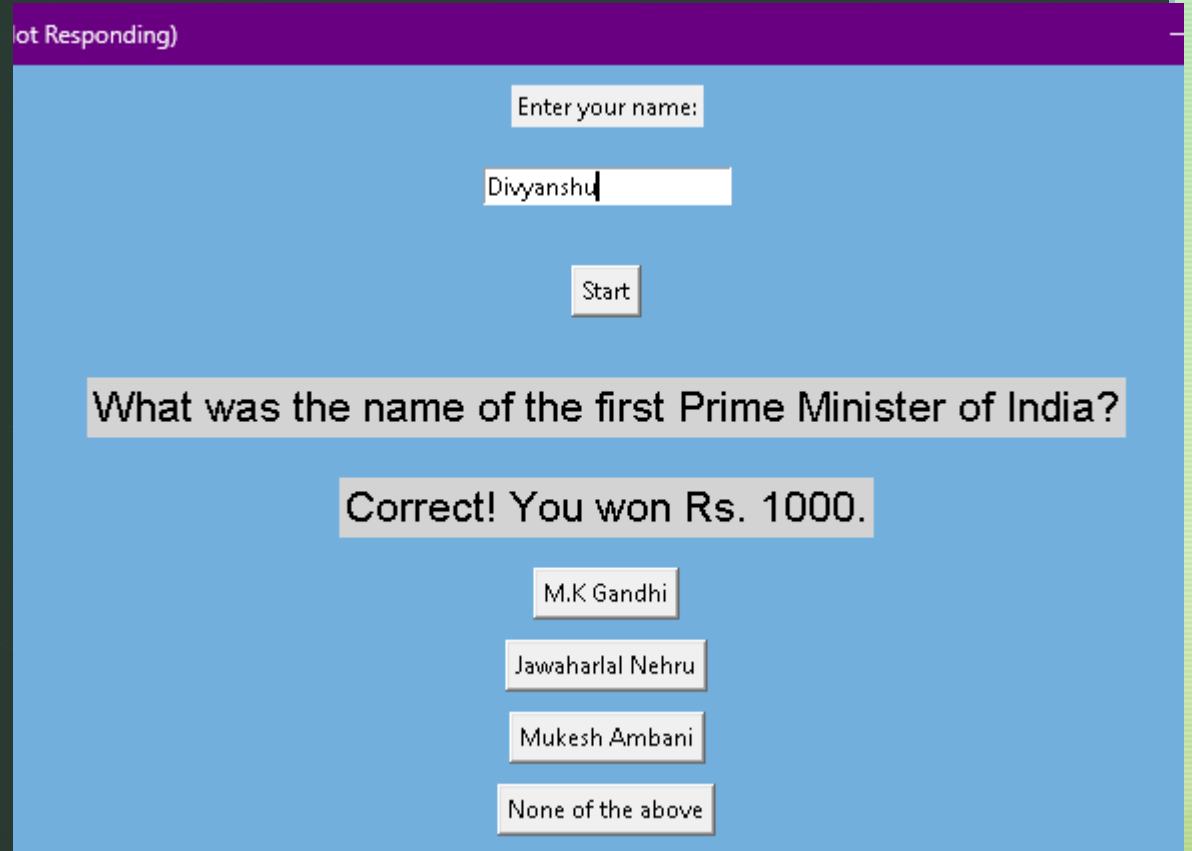


# SUMMARY :

The provided code successfully integrates multiple technologies to create an interactive quiz game with a user-friendly graphical interface, text-to-speech functionality, and audio feedback. By leveraging Python's tkinter for the GUI, win32com.client for text-to-speech, and pygame for audio playback, the game offers an engaging experience. Randomizing the question order with the random module ensures each game session is unique, adding replayability.

The game effectively handles user inputs through buttons, dynamically updates the GUI to reflect the player's progress, and offers immediate feedback via both visual elements and spoken text. The game state is carefully tracked, and both correct and incorrect answers result in appropriate actions, such as playing applause or ending the game.

This combination of audio-visual interaction, randomness, and a smooth game flow provides an entertaining way for users to test their knowledge while receiving immediate feedback on their answers. The modular design allows for easy extension, such as adding new questions or enhancing the audio-visual elements.



The screenshot displays a quiz application window. At the top, a purple status bar reads "(Not Responding)". Below this, a light blue main area contains a text input field labeled "Enter your name:" with the name "Divyanshu" entered. A "Start" button is positioned below the input field. The question "What was the name of the first Prime Minister of India?" is displayed in a white box. Below the question, a message box states "Correct! You won Rs. 1000.". At the bottom, four buttons are listed as options: "M.K Gandhi", "Jawaharlal Nehru", "Mukesh Ambani", and "None of the above".