import tkinter as tk

from tkinter import messagebox

import sqlite3

# Database setup

conn = sqlite3.connect("quiz\_app.db")

cursor = conn.cursor()

cursor.execute('''CREATE TABLE IF NOT EXISTS users (

id INTEGER PRIMARY KEY AUTOINCREMENT,

username TEXT UNIQUE,

password TEXT

)''')

conn.commit()

# Quiz questions

questions = [

{

"question": "What is the correct file extension for Python files?",

"answers": [".pyth", ".pt", ".py", ".pyt"],

"correct\_answer": 2

},

{

"question": "How do you create a variable with the numeric value 5 in Python?",

"answers": ["x = 5", "int x = 5", "num x = 5", "x : 5"],

"correct\_answer": 0

},

{

"question": "What is the correct syntax to output 'Hello World' in Python?",

"answers": ["echo 'Hello World'", "p('Hello World')", "print('Hello World')", "printf('Hello World')"],

"correct\_answer": 2

},

{

"question": "Which one of these is a mutable data type in Python?",

"answers": ["tuple", "string", "list", "list"],

"correct\_answer": 2

},

{

"question": "Which of the following keywords is used for function declaration in Python?",

"answers": ["function", "def", "func", "declare"],

"correct\_answer": 1

},

{

"question": "What is the output of 3 \* 'Python'?",

"answers": ["Python3", "Python Python Python", "Error", "3Python"],

"correct\_answer": 1

},

{

"question": "What is the output of len(['Python', 'Java', 'C++'])?",

"answers": ["2", "3", "4", "Error"],

"correct\_answer": 1

},

{

"question": "Which of the following is a Python framework for web development?",

"answers": ["React", "Django", "Spring", "Laravel"],

"correct\_answer": 1

},

{

"question": "How can you create a comment in Python?",

"answers": ["# This is a comment", "// This is a comment", "/\* This is a comment \*/", "-- This is a comment"],

"correct\_answer": 0

},

{

"question": "Which Python keyword is used to handle exceptions?",

"answers": ["except", "try", "catch", "throw"],

"correct\_answer": 1

}

]

# Initialize variables

score = 0

current\_question = 0

# Function to register a new user

def register\_user(username, password):

try:

cursor.execute("INSERT INTO users (username, password) VALUES (?, ?)", (username, password))

conn.commit()

messagebox.showinfo("Signup Success", "Account created successfully! Please login.")

show\_login\_screen()

except sqlite3.IntegrityError:

messagebox.showerror("Signup Error", "Username already exists! Please choose another.")

# Function to validate user login

def login\_user(username, password):

cursor.execute("SELECT \* FROM users WHERE username=? AND password=?", (username, password))

result = cursor.fetchone()

if result:

messagebox.showinfo("Login Success", f"Welcome, {username}!")

show\_quiz\_screen()

else:

messagebox.showerror("Login Error", "Invalid username or password!")

# Function to check answer and move to next question

def check\_answer(selected\_answer):

global score, current\_question

question\_data = questions[current\_question]

if selected\_answer == question\_data["correct\_answer"]:

score += 1

messagebox.showinfo("Result", "Correct!")

else:

correct\_answer\_text = question\_data["answers"][question\_data["correct\_answer"]]

messagebox.showinfo("Result", f"Wrong! The correct answer is: {correct\_answer\_text}")

current\_question += 1

if current\_question < len(questions):

show\_question(current\_question)

else:

messagebox.showinfo("Quiz Completed", f"Your final score is {score}/{len(questions)}.")

root.destroy()

# Function to display current question

def show\_question(index):

question\_data = questions[index]

question\_label.config(text=f"Question {index + 1}: {question\_data['question']}")

for i, answer in enumerate(question\_data["answers"]):

answer\_buttons[i].config(text=answer, command=lambda i=i: check\_answer(i))

# Function to display signup screen

def show\_signup\_screen():

clear\_window()

title\_label.config(text="Signup")

tk.Label(root, text="Username:", font=("Arial", 12)).pack(pady=5)

username\_entry.pack(pady=5)

tk.Label(root, text="Password:", font=("Arial", 12)).pack(pady=5)

password\_entry.pack(pady=5)

signup\_button = tk.Button(root, text="Signup", command=lambda: register\_user(username\_entry.get(), password\_entry.get()), font=("Arial", 12), bg="green", fg="white")

signup\_button.pack(pady=10)

switch\_to\_login\_button.config(text="Already have an account? Login", command=show\_login\_screen)

switch\_to\_login\_button.pack()

# Function to display login screen

def show\_login\_screen():

clear\_window()

title\_label.config(text="Login")

tk.Label(root, text="Username:", font=("Arial", 12)).pack(pady=5)

username\_entry.pack(pady=5)

tk.Label(root, text="Password:", font=("Arial", 12)).pack(pady=5)

password\_entry.pack(pady=5)

login\_button = tk.Button(root, text="Login", command=lambda: login\_user(username\_entry.get(), password\_entry.get()), font=("Arial", 12), bg="blue", fg="white")

login\_button.pack(pady=10)

switch\_to\_login\_button.config(text="Don't have an account? Signup", command=show\_signup\_screen)

switch\_to\_login\_button.pack()

# Function to display quiz screen after login

def show\_quiz\_screen():

clear\_window()

title\_label.pack\_forget() # Hide the title for the quiz screen

question\_label.pack(pady=20)

for btn in answer\_buttons:

btn.pack(pady=5)

show\_question(current\_question)

# Function to clear widgets for screen transitions

def clear\_window():

for widget in root.winfo\_children():

widget.pack\_forget()

# GUI Setup

root = tk.Tk()

root.title("Python Quiz Game")

root.geometry("500x400")

root.config(bg="#f0f0f0")

# Global widgets for reuse across screens

title\_label = tk.Label(root, text="Python Quiz Game", font=("Arial", 18, "bold"), bg="#f0f0f0", fg="black")

title\_label.pack(pady=20)

username\_entry = tk.Entry(root, font=("Arial", 12))

password\_entry = tk.Entry(root, show="\*", font=("Arial", 12))

switch\_to\_login\_button = tk.Button(root, text="", font=("Arial", 10), fg="blue", bg="#f0f0f0", borderwidth=0)

# Question and Answer widgets for the quiz screen

question\_label = tk.Label(root, text="", font=("Arial", 14), wraplength=400, justify="center", bg="#f0f0f0")

answer\_buttons = [tk.Button(root, text="", font=("Arial", 12), width=20, bg="#e0e0e0") for \_ in range(4)]

# Start with login screen

show\_login\_screen()

# Run the GUI

root.mainloop()

# Close the database connection when done

conn.close()