

LABORATORIO 3. PROYECTO FINAL

CATALINA ARCILA GRISALES.
ANGELA MARIA BOLAÑOS MONCAYO.
JUAN CAMILO POVEDA

Resumen- En el siguiente documento se presenta la implementación de un juego interactivo estilo “Trivia” desarrollada en Python, utilizando las bibliotecas Tkinter para la interfaz gráfica, PIL para el manejo de imágenes y JSON para almacenar. El juego permite crear el registro de usuarios, así como acceder a cada perfil inscrito que se almacena en el archivo “users.txt”. Al iniciar sesión el sistema utiliza el archivo “questions.txt” donde las preguntas están clasificadas por categorías como ciencia, música y deporte, las cuales se escogen aleatoriamente con un máximo de diez por cada sesión. Los usuarios responden las preguntas con opciones múltiples, ganando dos puntos por cada respuesta correcta y restando un punto por cada respuesta incorrecta. Al finalizar, el total de puntos acumulados de cada usuario se actualiza y se almacena. Finalmente, se analizan los conceptos aprendidos y aplicados, como el uso de clases para modelar y organizar el código en estructuras funcionales, manejo de memoria, y la importancia de un flujo lógico en el diseño de programas interactivos.

Palabras clave - Clases, objetos, Tkinter, JSON, Python.

I. INTRODUCCIÓN

En este documento se demuestra el uso de los conocimientos y habilidades adquiridas durante el proceso de aprendizaje en programación con el lenguaje Python, esto, para aplicarlo en el proyecto final, usando clases, programación orientada a objetos, creación de interfaces gráficas y el manejo de archivos, donde el docente, además se importan herramientas fundamentales como Tkinter [1], PIL [2] y JSON. Esto, con la finalidad de comprender y aprender a utilizar correctamente el lenguaje Python en este proyecto.

A continuación, se presenta la creación del juego al estilo de preguntados, donde se dará una explicación más profunda sobre su funcionamiento, además de la solución que se ofrece para

este juego, reflejando el uso y aprendizaje de lo enseñado en clases.

II. OBJETIVOS

Objetivo general:

Diseñar e implementar un juego interactivo, basado en un juego de preguntas y respuestas que permita a los usuarios participar y adquirir nuevos conocimientos de manera entretenida.

Objetivos específicos:

- Añadir una base de preguntas categorizadas las cuales se seleccionan y presentan de forma aleatoria en cada sesión del juego.
- Realizar un sistema de puntuaciones dinámica que motive a los usuarios a mejorar su interacción en el juego.
- Asegurar que el juego que sea fácil de usar, con sus respectivas funciones.

III. MÉTODOS E INSTRUMENTOS

El lenguaje de programación empleado fue Python a través de Visual Studio Code, para crear y ejecutar el código, la interfaz gráfica, como las imágenes de fondo, se utilizaron para personalizar el juego y brindar una mejor experiencia.

```
1 from tkinter import *
2 from PIL import Image, ImageTk
3 from json import load, dump, JSONDecodeError
4 from random import shuffle
5 from tkinter import messagebox
6
7 # Clase para manejar usuarios
8 class UserManager:
9     def __init__(self, file_name="users.txt"):
10         self.file_name = file_name
11         self.users = self.load_users()
12
13     def load_users(self):
14         try:
15             with open(self.file_name, "r") as file:
16                 return load(file)
17         except (FileNotFoundException, JSONDecodeError):
18             return []
19
20     def save_users(self):
21         with open(self.file_name, "w") as file:
22             dump(self.users, file, indent=4)
23
24     def register_user(self, name, password):
25         if anyuser(name) == name for user in self.users:
26             return False
27         new_user = {"name": name, "password": password, "active": 0, "score": 0}
28         self.users.append(new_user)
29         self.save_users()
30         return True
31
32     def login_user(self, name, password):
```

```

12 for user in self.users:
13     if user['name'] == name and user['password'] == password:
14         user['active'] = 1
15         self.save_users()
16         return user
17     return None
18
19 def update_user(self, user):
20     for idx, u in enumerate(self.users):
21         if u['name'] == user['name']:
22             self.users[idx] = user
23             self.save_users()
24             break
25
26 # Clase para manejar las preguntas
27 class QuestionManager:
28     def __init__(self, file_name="questions.txt"):
29         self.file_name = file_name
30         self.questions = self.load_questions()
31
32     def load_questions(self):
33         try:
34             with open(self.file_name, "r", encoding="utf-8") as file:
35                 data = json.load(file)
36                 all_questions = []
37                 for category, questions in data.items():
38                     for question in questions:
39                         question['category'] = category
40                         all_questions.append(question)
41
42         except FileNotFoundError:
43             return []
44
45 # Clase principal del juego
46 class TriviaApp(tk.Tk):
47     def __init__(self):
48         super().__init__()
49         self.title("Trivia App")
50         self.state("normal")
51         self.user_manager = UserManager()
52         self.question_manager = QuestionManager()
53
54         self.current_user = None
55         self.current_question_index = 0
56         self.puntuacion = 0
57
58         self.original_image = image.open("1.png")
59         screen_width = self.winfo_screenwidth()
60         screen_height = self.winfo_screenheight()
61         self.bg_image = image.open(self.original_image.resized((screen_width, screen_height)))
62         self.bg_label = Label(self, image=self.bg_image)
63         self.bg_label.place(x=0, y=0, relwidth=1, relheight=1)
64
65         Button(self, text="Comenzar", font=("Arial", 20), command=self.start, place(relx=0.5, rely=0.55, anchor="center"))
66
67     def clear_widgets(self):
68         # Elimina todos los widgets de la ventana
69         for widget in self.winfo_children():
70             widget.destroy()
71
72     def start(self):
73         self.clear_widgets()
74         self.main_menu()
75
76     def main_menu(self):
77         self.load_background("1.png")
78
79         Label(self, text="Bienvenido a Trivia App", font=("Arial", 24), bg="darkorange", pack(pady=10))
80
81         Button(self, text="Registrar Usuario", font=("Arial", 18), command=self.register_user, place(relx=0.5, rely=0.717, anchor="center"))
82         Button(self, text="Iniciar Sesión", font=("Arial", 18), command=self.open_session, place(relx=0.5, rely=0.81, anchor="center"))
83
84     def register_user(self):
85         # Pantalla para registrar
86         self.clear_widgets()
87         self.load_background("2.png")
88
89         Label(self, text="Nombre:", font=("Arial", 16), bg="gray", fg="white", place(relx=0.5, rely=0.5, anchor="center"))
90         entry_name = Entry(self, font=("Arial", 12), bg="gray", fg="white")
91         entry_name.place(relx=0.5, rely=0.55, anchor="center")
92
93         Label(self, text="Contraseña:", font=("Arial", 16), bg="gray", fg="white", place(relx=0.5, rely=0.7, anchor="center"))
94         entry_password = Entry(self, show="*", font=("Arial", 12), bg="gray", fg="white")
95         entry_password.place(relx=0.5, rely=0.75, anchor="center")
96
97         def login_user():
98             name = entry_name.get()
99             password = entry_password.get()
100             user = self.user_manager.login_user(name, password)
101             if user:
102                 self.current_user = user
103                 messagebox.showinfo("Inicio", "Sesión iniciada como (%s)" % name)
104                 self.main_menu()
105             else:
106                 messagebox.showerror("Error", "Credenciales incorrectas.")
107
108         Button(self, text="Iniciar Sesión", font=("Arial", 18), bg="lightblue", fg="white", command=login_user, place(relx=0.5, rely=0.85, anchor="center"))
109
110     def open_session(self):
111         # Pantalla para iniciar sesión
112         self.clear_widgets()
113         self.load_background("2.png")
114
115         Label(self, text="Nombre:", font=("Arial", 16), bg="gray", fg="white", place(relx=0.5, rely=0.5, anchor="center"))
116         entry_name = Entry(self, font=("Arial", 12), bg="gray", fg="white")
117         entry_name.place(relx=0.5, rely=0.55, anchor="center")
118
119         Label(self, text="Contraseña:", font=("Arial", 16), bg="gray", fg="white", place(relx=0.5, rely=0.7, anchor="center"))
120         entry_password = Entry(self, show="*", font=("Arial", 12), bg="gray", fg="white")
121         entry_password.place(relx=0.5, rely=0.75, anchor="center")
122
123     def login_user(self):
124         name = entry_name.get()
125         password = entry_password.get()
126         user = self.user_manager.login_user(name, password)
127         if user:
128             self.current_user = user
129             messagebox.showinfo("Inicio", "Sesión iniciada como (%s)" % name)
130             self.main_menu()
131         else:
132             messagebox.showerror("Error", "Credenciales incorrectas.")
133
134     def load_background(self, image_path):
135         self.original_image = image.open(image_path)
136         screen_width = self.winfo_screenwidth()
137         screen_height = self.winfo_screenheight()
138         self.bg_image = image.open(self.original_image.resized((screen_width, screen_height)))
139         self.bg_label = Label(self, image=self.bg_image)
140         self.bg_label.place(x=0, y=0, relwidth=1, relheight=1)
141
142     def user_menu(self):
143         self.clear_widgets()
144         self.load_background("4.png")
145
146         Label(self, text="Bienvenido, (%s)" % self.current_user['name'], font=("Arial", 24), bg="pale violet red", place(relx=0.5, rely=0.27, anchor="center"))
147
148         users_frame = Frame(self, bg="lightgreen", width=600, height=100)
149         users_frame.place(relx=0.5, rely=0.35, anchor="center")
150
151         Label(users_frame, text="Usuarios Conectados", font=("Arial", 14), bg="lightgreen", fg="black", pack(pady=10))
152
153         # Mostrar los usuarios conectados y sus puntajes
154         for user in self.user_manager.users:

```

```

155         if user['active']:
156             user_info = "Nombre: (%s) | Puntaje: (%s)" % (user['name'], user['score'])
157             Label(users_frame, text=user_info, font=("Arial", 12), bg="lightgreen", fg="black", pack(pady=5))
158
159     def user_puntaje(self):
160         command=self.view_score, font=("Arial", 14), width=20, place(relx=0.5, rely=0.4, anchor="center")
161     def iniciar_preguntas(self):
162         command=self.start_qt, font=("Arial", 14), width=20, place(relx=0.5, rely=0.5, anchor="center")
163     def cerrar_sesion(self):
164         command=self.logout, font=("Arial", 14), width=20, place(relx=0.5, rely=0.6, anchor="center")
165
166     def view_score(self):
167         if self.current_user:
168             messagebox.showinfo("Puntaje", "Tu puntaje es: (%s)" % (self.current_user['score']))
169
170     def start_qt(self):
171         self.puntuacion_ronda = 0
172         self.current_question_index = 0
173
174         if not self.question_manager.questions:
175             messagebox.showerror("Error", "No hay preguntas disponibles.")
176             return
177
178         self.show_question()
179
180     def show_question(self):
181         self.current_question_index > len(self.question_manager.questions):
182             self.show_results()
183             return
184
185         self.clear_widgets()
186         self.load_background("5.png")
187
188     def pregunta(self):
189         pregunta = self.question_manager.questions[self.current_question_index]
190         Label(self, text="Categoría: (%s)" % pregunta['categoria'], font=("Arial", 16), bg="coral", pack(pady=20))
191         Label(self, text=pregunta['pregunta'], font=("Arial", 16), bg="white", wraplength=700, justify="center", bg="gray", pack(pady=20))
192
193         self.score_label = Label(self, text="Puntaje actual: (%s)" % (self.puntuacion_ronda), font=("Arial", 16), bg="gray", fg="white")
194         self.score_label.pack(pady=40)
195
196         options_frame = Frame(self, bg="gray")
197         options_frame.pack(pady=10)
198
199         for option, texto in pregunta['opciones'].items():
200             Button(options_frame, text=option, width=25, command=lambda option=option: self.check_answer(option), pack(pady=20))
201
202     def check_answer(self, respuesta_usuario):
203         pregunta = self.question_manager.questions[self.current_question_index]
204
205         if respuesta_usuario == pregunta['respuesta_correcta']:
206             self.puntuacion_ronda += 1
207         else:
208             self.puntuacion_ronda += 1
209
210         self.current_question_index += 1
211         self.show_question()
212
213     def show_results(self):
214         self.clear_widgets()
215         self.load_background("7.png")
216
217         # Actualizar el puntaje acumulado
218         if self.current_user:
219             self.current_user['score'] += self.puntuacion_ronda
220             self.user_manager.update_user(self.current_user)
221
222         Label(self, text="Puntuación final: (%s)" % (self.puntuacion_ronda), font=("Arial", 20), bg="coral", fg="white", place(relx=0.5, rely=0.5, anchor="center"))
223         Button(self, text="Volver al Inicio", command=self.main_menu, width=20, font=("Arial", 20), bg="gray", fg="white", place(relx=0.5, rely=0.6, anchor="center"))
224
225     def login(self):
226         self.current_user = None
227         self.main_menu()
228
229 app = TriviaApp()
230 app.mainloop()

```

Fig. 1. Código en Python.

El código implementa tres clases principales que configuran la funcionalidad del juego:

1. **UserManager:** Esta clase contiene toda la funcionalidad que tiene que ver con los usuarios. Implementa el registro, inicio de sesión y la actualización de datos. Empezando con la función:
 - load_user(self) línea 13, carga los datos de usuario desde el archivo users.txt. Si no existe o este vacío, retorna a una lista vacía.
 - register_user(self, name, password) línea 24, verifica si el nombre del usuario ya existe. Si no registra un nuevo usuario con su nombre y contraseña, estado y puntaje.
 - login_user(self, name, password) línea 32, comprueba las credenciales de inicio de sesión, si son correctas, marca al usuario como activo.
 - update_user(self, user) línea 40, actualiza los datos de un usuario y los guarda en el archivo.
2. **QuestionManager:** Implementa las preguntas del juego
 - load_questions(self) línea 53 carga las preguntas desde el archivo question.txt, las organiza por categorías y selecciona máximo diez preguntas aleatorias.
3. **TriviaApp:** Crea la interfaz gráfica.

-main_menu(self) línea 102, muestra el menú principal con opciones para registrar nuevos usuarios, iniciar sesión y acceder al juego.

-start_quiz(self) línea 194, inicia el juego, reestablece el puntaje de cada ronda y muestra la primera pregunta.

-show_question(self) línea 204, muestra una pregunta con sus opciones y permite al usuario seleccionar la respuesta.

-check_answer(self, respuesta_usuario) línea 227, revisa si la respuesta que ha seleccionado el usuario es correcta o incorrecta, actualizando el puntaje.

-show_results(self) línea 238, muestra el puntaje que ha alcanza el usuario y lo guarda en el perfil del mismo usuario.

-logout(self) línea 250, finaliza la sesión del usuario actual y vuelve al menú principal.

Archivos:

```
1 users.txt
2 {
3   "name": "mimi",
4   "password": "mimi",
5   "active": 0,
6   "score": -25
7 },
8 {
9   "name": "diana",
10  "password": "1701",
11  "active": 0,
12  "score": 4
13 },
14 {
15   "name": "Isabella ",
16   "password": "1011",
17   "active": 0,
18   "score": 10
19 },
20 {
21   "name": "Angela",
22   "password": "321",
23   "active": 1,
24   "score": 28
25 }
26 }
```

Fig. 2. Archivo (users.txt) que almacena al usuario.

```
1 preguntas.txt
2 {
3   "Ciencia": [{"pregunta": "¿Cuántos huesos hay en el cuerpo humano?", "opciones": ["A", "206", "B", "202", "C", "144", "D", "106"], "respuesta": "A"}, {"pregunta": "¿Qué misión lunar fue la primera en llevar un vehículo lunar?", "opciones": ["A", "Misión Apollo 11", "B", "Misión Apollo 8", "C", "Misión Apollo 13", "D", "Misión Apollo 16"], "respuesta": "A"}, {"pregunta": "¿A qué temperatura el agua se evapora?", "opciones": ["A", "80 grados", "B", "100 grados", "C", "110 grados", "D", "120 grados"], "respuesta": "B"}, {"pregunta": "¿En qué planeta se encuentra la montaña volcánica Olympus Mons?", "opciones": ["A", "Marte", "B", "Júpiter", "C", "Saturno", "D", "Neptuno"], "respuesta": "A"}, {"pregunta": "¿Cuál es el proceso mediante el cual las plantas convierten la energía luminosa en energía química?", "opciones": ["A", "Fotosíntesis", "B", "Respiración celular", "C", "Fermentación", "D", "Oxidación", "respuesta correcta": "A"}, {"pregunta": "¿Cuál es el órgano más grande del cuerpo humano?", "opciones": ["A", "Piel", "B", "Intestino grueso", "C", "Intestino delgado", "D", "Hígado", "respuesta correcta": "A"}, {"pregunta": "¿Cuál es el símbolo químico del oro?", "opciones": ["A", "Au", "B", "Ag", "C", "Zn", "D", "Fe"], "respuesta correcta": "A"}, {"pregunta": "¿Cómo se llama el planeta más pequeño de nuestro sistema solar?", "opciones": ["A", "Mercurio", "B", "Venus", "C", "Tierra", "D", "Marte", "respuesta correcta": "A"}, {"pregunta": "¿Cómo se llama el océano más grande del mundo?", "opciones": ["A", "Atlántico", "B", "Índico", "C", "Pacífico", "D", "Ártico", "respuesta correcta": "C"}, {"pregunta": "¿La Tierra gira sobre su eje una vez cada?", "opciones": ["A", "364 días", "B", "24 horas", "C", "12 horas", "D", "120 días"], "respuesta correcta": "B"}, {"pregunta": "¿Qué cantante es considerado el rey del rock and roll?", "opciones": ["A", "Elvis Presley", "B", "Chuck Berry", "C", "The Beatles", "D", "The Rolling Stones", "respuesta correcta": "A"}, {"pregunta": "¿Quién compuso 'El lago de los cisnes' y 'El cascanueces'?", "opciones": ["A", "Ludwig van Beethoven", "B", "Piotr Ilich Chaikovski", "C", "Franz Schubert", "D", "Johann Sebastian Bach", "respuesta correcta": "A"}, {"pregunta": "¿Qué músico compositor fue el creador de la ópera 'Los boleros de España'?", "opciones": ["A", "Luis de Buñuel", "B", "Joaquín Turina", "C", "Manuel de Falla", "D", "Isaac Albéniz", "respuesta correcta": "C"}, {"pregunta": "¿De qué nacionalidad era Beethoven?", "opciones": ["A", "Alemana", "B", "Austriaca", "C", "Francesa", "D", "Italiana", "respuesta correcta": "A"}, {"pregunta": "¿Qué cantante es reconocido por éxitos como 'Thriller', 'Beat It' y 'Smells Like Teen Spirit'?", "opciones": ["A", "Prince", "B", "Michael Jackson", "C", "Madonna", "D", "Beyoncé", "respuesta correcta": "B"}, {"pregunta": "¿Cuántos cuartos tiene un violín?", "opciones": ["A", "Tres", "B", "Cuatro", "C", "Cinco", "D", "Seis", "respuesta correcta": "B"}, {"pregunta": "¿Qué compositor y director de orquesta italiano fue profesor de músicos como Beethoven y Schubert?", "opciones": ["A", "Antonio Vivaldi", "B", "Wolfgang Amadeus Mozart", "C", "Ludwig van Beethoven", "D", "Franz Schubert", "respuesta correcta": "A"}, {"pregunta": "¿Qué banda fundó el rock?", "opciones": ["A", "Metallica", "B", "The Beatles", "C", "The Rolling Stones", "D", "The Who", "respuesta correcta": "B"}, {"pregunta": "¿Cómo se escribe solo en el idioma americano?", "opciones": ["A", "Solo", "B", "Solos", "C", "Sol", "D", "Solos", "respuesta correcta": "A"}, {"pregunta": "¿Quién es el líder de la banda de rock Queen?", "opciones": ["A", "Bryan Adams", "B", "Sebastian Bach", "C", "James Hetfield", "D", "Freddie Mercury", "respuesta correcta": "D"}, {"pregunta": "¿Cuál es la mano más alta posible en el póker?", "opciones": ["A", "Full house", "B", "Una escalera", "C", "Una escalera y un par", "D", "Una escalera y un par", "respuesta correcta": "A"}, {"pregunta": "¿Qué equipo ganó la primera Super Bowl de la historia?", "opciones": ["A", "Los New England Patriots", "B", "Los Pittsburgh Steelers", "C", "Los Dallas Cowboys", "D", "Los Baltimore Colts", "respuesta correcta": "A"}, {"pregunta": "¿Qué se inventó primero: las damas o el ajedrez?", "opciones": ["A", "Las damas", "B", "El ajedrez", "C", "Ambos al mismo tiempo", "D", "Ninguno", "respuesta correcta": "B"}, {"pregunta": "¿Cuántos kilómetros tiene una maratón?", "opciones": ["A", "42 km", "B", "40 km", "C", "45 km", "D", "50 km", "respuesta correcta": "A"}, {"pregunta": "¿Qué color de ciudad reciben los alumnos que empiezan a aprender karate?", "opciones": ["A", "Rojo", "B", "Naranja", "C", "Blanco", "D", "Negro", "respuesta correcta": "C"}, {"pregunta": "¿Este deporte se juega con una red en un patio?", "opciones": ["A", "Fútbol", "B", "Vóley", "C", "Bádminton", "D", "Tenis", "respuesta correcta": "D"}, {"pregunta": "¿Qué tres deportes componen un triatlón?", "opciones": ["A", "Carrera, ciclismo y golf", "B", "Carrera, natación y ciclismo", "C", "Natación, ciclismo y golf", "D", "Carrera, natación y fútbol", "respuesta correcta": "B"}, {"pregunta": "¿Cuál es el deporte más practicado (y visto) en el mundo?", "opciones": ["A", "El fútbol", "B", "El cricket", "C", "El béisbol", "D", "El baloncesto", "respuesta correcta": "A"}, {"pregunta": "¿Cuál de los siguientes deportes es un deporte acuático?", "opciones": ["A", "Fútbol", "B", "Tenis", "C", "Vóley", "D", "Baloncesto", "respuesta correcta": "D"}, {"pregunta": "¿Qué ciudad ha albergado más Juegos Olímpicos de verano?", "opciones": ["A", "París", "B", "Londres", "C", "Roma", "D", "Los Ángeles", "respuesta correcta": "A"}]
```

Fig. 2.1. Archivo con preguntas de ciencia, música y deportes.

Pruebas:

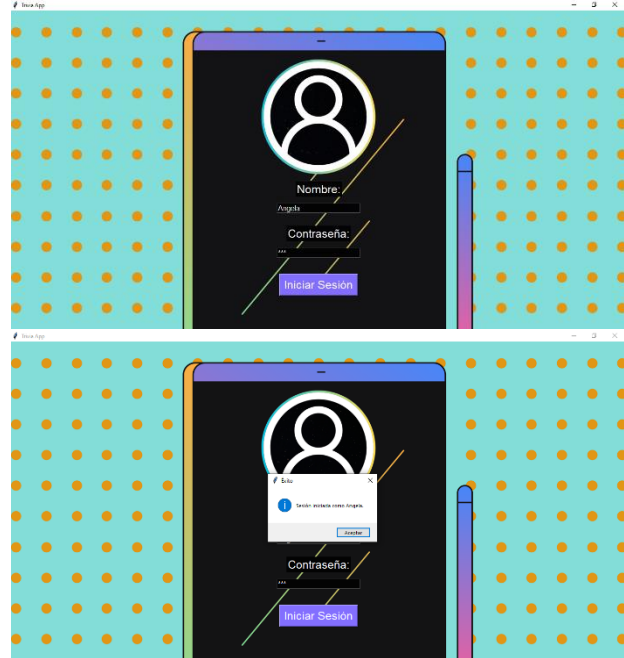


Fig. 3. Iniciar sesión.

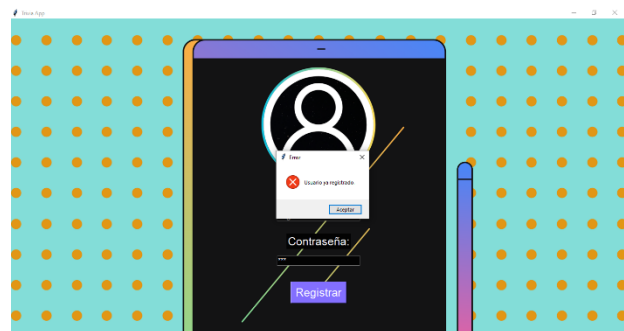


Fig. 3.1. Registrar un usuario que ya está registrado.

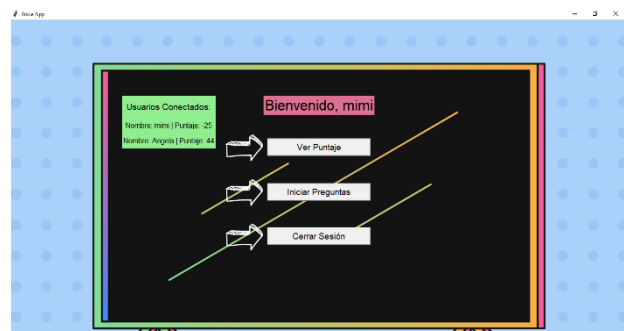


Fig. 3.2. Menú principal y usuarios conectados.



Fig. 3.3. Preguntas y respuestas de selección múltiple.

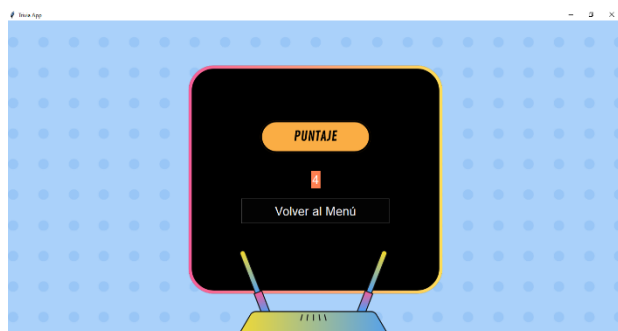


Fig. 3.3. Finalización del juego.

Uso de memoria

El código emplea la memoria para gestionar los datos de usuarios, preguntas y elementos de la interfaz gráfica. Los usuarios y preguntas se importan desde archivos JSON y se almacenan en listas de memoria mientras se ejecuta el programa. La interfaz gráfica, creada mediante Tkinter [1], utiliza memoria para mantener activos widgets e imágenes, liberándolos cuando no son necesarios mediante funciones como `clear_window`. Por lo tanto, las imágenes de alta resolución y grandes las listas de preguntas son los principales consumidores de memoria.

CONCLUSIONES

1. El uso de bibliotecas y módulos como Tkinter, PIL y `messagebox` fueron fundamentales para la creación y funcionalidad de la interfaz gráfica del juego Trivia. Su implementación representó un reto, ya que eran completamente nuevas, lo que requirió investigar su funcionamiento y manejo para integrarlas adecuadamente.
2. Se destacó la investigación sobre el uso de botones para mejorar la experiencia del usuario, buscando que estos

fueran intuitivos y que realizaran acciones claras dentro del juego. Además, se añadieron bloques de textos que explicaban la funcionalidad de cada botón [3], facilitando la interacción.

3. Uno de los mayores desafíos fue implementar una ruleta para seleccionar aleatoriamente la categoría de las preguntas. Aunque se intentó resolver, no se implementó al no tener éxito, gracias a esto se contemplaron otras alternativas que permitieron mantener la lógica y el entretenimiento del juego.
4. Por último, este laboratorio permitió afianzar conocimientos en Python, tanto básicos como avanzados. Se aprendió a manejar clases y programación orientada a objetos, además de integrar conceptos como categorías y constructores para la base del juego, lo que brindó un conocimiento más amplio sobre programación y diseño de interfaces gráficas.

REFERENCIAS

- [1] “tkinter.messagebox — Indicadores de mensajes de Tkinter” — documentación de Python - 3.10.15”, *Python.org*. [En línea]. Disponible en: <https://docs.python.org/es/3.10/library/tkinter.messagebox.html>. [Consultado: 22-nov-2024].
- [2] “Pillow”, *Pillow (PIL Fork)*. [En línea]. Disponible en: <https://pillow.readthedocs.io/en/stable/>. [Consultado: 22-nov-2024].
- [3] A. López-Parrado, *tkinter/test_button_click.py at main · parrado/lecture2-code-G1*. .