Documentación del Proyecto: Juego de Trivia Web

Objetivo del Proyecto

Este proyecto consiste en el desarrollo de una aplicación web interactiva de trivia, en la cual los usuarios pueden iniciar sesión, seleccionar una categoría temática, responder preguntas contrarreloj y acumular puntajes. Los resultados se guardan en un sistema de ranking visible desde la pantalla principal. El objetivo es combinar conocimientos de HTML, CSS, JavaScript, Python y Flask en una experiencia web funcional, estética y dinámica.

Tecnologías Utilizadas

HTML5	Estructura del contenido de las páginas (login, main, trivia, etc.)
CSS / Bootstrap	Estilizado personalizado de la interfaz, en combinación con Bootstrap
JavaScript	Lógica en el navegador: validaciones, control del flujo del juego, cronómetro
Python (Flask)	Framework de Python utilizado como servidor web y motor de plantillas
SQLite	Base de datos liviana, utilizada para guardar y recuperar los puntajes del ranking

Estructura del Proyecto

El proyecto sigue una estructura organizada basada en los estándares de Flask:

```
/Proyecto/
├── /static/          # Archivos estáticos públicos
│   ├── /css/         # Hojas de estilo (main.css, login.css, trivia.css)
│   ├── /js/          # Lógica del juego, validaciones y navegación
│   ├── /img/         # Imágenes usadas en la UI (íconos, fondos)
│   └── /sounds/      # Efectos de sonido (temporizador, etc.)
```

```
├── /templates/          # Archivos HTML renderizados con Jinja2
│   ├── login.html
│   ├── main.html
│   └── trivia.html
├── app.py               # Archivo principal del servidor Flask
├── database.py          # Funciones para interactuar con la base de datos
└── ranking.db           # Base de datos SQLite con los puntajes
```

Flujo del Usuario

- • Inicio de sesión:

El usuario accede a un formulario de login donde debe ingresar su nombre de usuario y una contraseña. El nombre es validado con JavaScript y, si es correcto, el sistema guarda el nombre en una sesión de Flask.

- • Pantalla principal (main.html):

Aparece un saludo personalizado con el nombre del jugador. Se muestran los puntajes acumulados por cada jugador en un ranking general (ordenado de mayor a menor). Desde acá se puede iniciar una nueva partida.

- • Juego de trivia (trivia.html):

El usuario elige una categoría de preguntas. Cada categoría tiene 3 preguntas aleatorias con límite de tiempo de 30 segundos. Se utiliza un contador visual y un sonido de fondo.

- • Pantalla de resultados:

Al finalizar la trivia, se muestra el puntaje obtenido. Este se guarda en la base de datos y se suma al puntaje anterior del jugador.

- • Retorno al menú principal:

Se actualiza el ranking y se puede volver a jugar.

Lógica del Ranking (Base de Datos)

La base de datos SQLite contiene una única tabla llamada ranking, que almacena el nombre del jugador, su puntaje y la fecha en que jugó:

```
CREATE TABLE ranking (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  nombre TEXT NOT NULL,
```

```
    puntaje INTEGER NOT NULL,
    fecha TEXT DEFAULT CURRENT_TIMESTAMP
);
```

Al insertar un puntaje, si el nombre del jugador ya existe, se actualiza su puntaje sumando el nuevo. Si no existe, se crea una nueva entrada:

```
def insertar_puntaje(nombre, puntaje):
    with sqlite3.connect("ranking.db") as conn:
        cursor = conn.cursor()
        cursor.execute("SELECT puntaje FROM ranking WHERE nombre
= ?", (nombre,))
        resultado = cursor.fetchone()

        if resultado:
            nuevo_puntaje = resultado[0] + puntaje
            cursor.execute("UPDATE ranking SET puntaje = ? WHERE
nombre = ?", (nuevo_puntaje, nombre))
        else:
            cursor.execute("INSERT INTO ranking (nombre, puntaje,
fecha) VALUES (?, ?, datetime('now'))", (nombre, puntaje))

        conn.commit()
```

Funcionalidades Destacadas

- • Validaciones de formulario con íconos y mensajes personalizados.
- • Contador de tiempo con sonido en cada pregunta.
- • Sistema de pantallas dinámicas controladas por JavaScript.
- • Sistema de puntajes acumulativos persistente en base de datos.
- • Interfaz clara, responsiva y accesible con Bootstrap.

Conclusión

Este proyecto nos permitió aplicar lo aprendido durante la cursada de una forma completa y práctica. Pudimos integrar tecnologías del front-end y back-end, trabajar con sesiones de usuario, bases de datos y manejar interacciones complejas en el navegador. Además, logramos una experiencia de usuario agradable, visualmente atractiva y funcional.

Queda como posibilidad para futuros desarrollos agregar niveles de dificultad, preguntas desde una API externa, o incluso modos multijugador.