1. INICIALIZAR JUEGO

Declaraciones e Inicializaciones:

- Tablero: Una estructura de datos (puede ser un diccionario) donde las claves son las categorías ("Marca", "Comida", "Lugar", "Animal") y los valores son otro diccionario que contiene la palabra y el estado de bloqueo ({word: "", locked: False}).
- Letra Aleatoria: Variable que almacenará una letra generada aleatoriamente del alfabeto inglés.
- Tiempo de Juego: Un temporizador que controlará la duración máxima del juego, por ejemplo, un minuto desde que comienza.

Elementos a Considerar:

- La función para generar la letra aleatoria debe ser capaz de seleccionar de forma uniforme entre todas las letras del alfabeto.
- Asegurarse de que el tablero esté correctamente inicializado sin palabras y con todas las categorías desbloqueadas.

2. AL UNIRSE UN JUGADOR

Declaraciones:

- Cuando un jugador se une al juego, se le debe mostrar el estado actual del tablero y la letra aleatoria.

Elementos a Considerar:

- Gestionar la conexión del jugador al servidor, usando sockets.
- Asegurarse de que cada jugador recibe la información correcta y actualizada del estado del juego al momento de su conexión.

3. CUANDO UN JUGADOR QUIERE ESCRIBIR EN UNA CATEGORÍA

Declaraciones e Inicializaciones:

- **Control de Acceso:** Antes de que un jugador pueda escribir en una categoría, debe verificar si esta está "libre". Si lo está, el jugador "coge la bandera" (bloquea la categoría), escribe su palabra, y luego "suelta la bandera" (desbloquea la categoría tras un breve periodo, por ejemplo, 5 segundos).

Elementos a Considerar:

- Implementar mecanismos de bloqueo para evitar que múltiples jugadores modifiquen la misma categoría simultáneamente.
- Verificar que las palabras ingresadas comiencen con la letra aleatoria designada.
- Implementar una forma de notificar a todos los jugadores sobre las actualizaciones del tablero.

4. DURANTE EL JUEGO

Declaraciones:

- Permitir que los jugadores interactúen con el tablero mientras el tiempo del juego no haya expirado.

Elementos a Considerar:

- Necesidad de un bucle o monitor que constantemente revise si todas las categorías están llenas o si el tiempo de juego ha expirado.
- Debe haber un mecanismo para finalizar el juego automáticamente cuando se cumplan las condiciones de terminación.

5. AL TERMINAR EL TIEMPO DE JUEGO O COMPLETAR EL TABLERO

Declaraciones:

- Finalización del juego, mostrando el tablero final a todos los jugadores.
- Liberación de todos los recursos, especialmente desbloqueando cualquier categoría que aún esté bloqueada.

Elementos a Considerar:

- Asegurar que el juego se detenga correctamente y que todos los recursos (como categorías bloqueadas) sean liberados.
- Notificar a todos los jugadores del resultado final y asegurarse de que la comunicación del final del juego sea clara.

Importante cuando la lógica se encuentra en el servidor

Actualización del Tablero: Cuando un jugador envía una palabra para una categoría, el servidor primero verifica si la categoría está desbloqueada. Si es así, actualiza la palabra, establece la categoría como bloqueada, y luego, después de un breve período (por ejemplo, 5 segundos), la desbloquea automáticamente

Sincronización y Comunicación: El servidor utiliza tecnologías como WebSocket para comunicar en tiempo real las actualizaciones del tablero a todos los jugadores conectados, asegurando que todos vean el estado actualizado del juego de forma simultánea.

```
socketio = SocketIO(app)

def notify_players(game_id):
    socketio.emit('board_update', {'board': dict(tablero)}, room=game_id)
```

PSEUDOCÓDIGO CON LAS REGLAS DEL JUEGO

INICIALIZAR JUEGO:

crear tablero con categorías como {Marca, Comida, Lugar, Animal} cada una libre

generar letra aleatoria

establecer tiempo de juego (ej: 1 minuto)

esperar a que los jugadores se unan

AL UNIRSE UN JUGADOR:

mostrar al jugador la letra aleatoria y el estado actual del tablero

CUANDO UN JUGADOR QUIERE ESCRIBIR EN UNA CATEGORÍA:

si la categoría está libre:

"coger la bandera" (bloquear la categoría)

permitir al jugador escribir una palabra que comience con la letra aleatoria

actualizar el tablero con la palabra del jugador

"soltar la bandera" (desbloquear la categoría después de 5 segundos)

notificar a todos los jugadores el cambio en el tablero

si la categoría está bloqueada:

informar al jugador que la categoría está en uso y pedir que intente otra

DURANTE EL JUEGO:

permitir a los jugadores escribir en categorías libres siguiendo el procedimiento de bloqueo

revisar si todas las categorías están llenas:

si están llenas, terminar el juego y declarar el tablero completo

si no, continuar hasta que el tiempo de juego expire

AL TERMINAR EL TIEMPO DE JUEGO O COMPLETAR EL TABLERO:

finalizar el juego

mostrar a todos los jugadores el tablero final

liberar todos los recursos (categorías bloqueadas)

PARTE DE CÓDIGO PARA INICIALIZAR EL JUEGO

```
import random

import string

from collections import defaultdict

from threading import Timer

#En esTa configuración, el servidor mantiene el estado completo del tablero,
#maneja las actualizaciones de este estado según las acciones de los
#jugadores, y se encarga de la sincronización y la lógica del juego

# Creación del tablero

tablero = defaultdict(lambda: {'word': "", 'locked': False})

categorias = ["Marca", "Comida", "Lugar", "Animal"]


# Generación de letra aleatoria

letra_aleatoria = random.choice(string.ascii_uppercase)


# Configuración del tiempo de juego

tiempo_de_juego = 60  # Tiempo en segundos

def finalizar_juego():

    print("El juego ha finalizado.")

    # Aquí iría cualquier limpieza o lógica de finalización del juego

temporizador = Timer(tiempo_de_juego, finalizar_juego)
```

SI TENÉIS EL TABLERO EN LA API (CONTROLA EL ESTADO DEL TABLERO)

```
def obtener_estado_tablero(game_id):

    response = requests.get(f"https://api.juego.com/tablero/{game_id}")

    return response.json()


def enviar_palabra(game_id, categoria, palabra):

    data = {'categoria': categoria, 'palabra': palabra}

    response = requests.post(f"https://api.juego.com/tablero/{game_id}/actualizar",
json=data)

    return response.json()
```

Escenarios de Uso de APIs en el Juego de STOP

```
def obtener_estado_tablero(game_id):
    response = requests.get(f"https://api.juego.com/tablero/{game_id}")
    return response.json()


def enviar_palabra(game_id, categoria, palabra):
    data = {'categoria': categoria, 'palabra': palabra}
    response = requests.post(f"https://api.juego.com/tablero/{game_id}/actualizar", json=data)
    return response.json()


def validar_palabra(palabra, letra):
    response = requests.get(f"https://api.diccionario.com/validar?palabra={palabra}&letra={letra}")
    es_valida = response.json().get('es_valida')
    return es_valida
def actualizar_puntuacion(usuario_id, puntos):
    requests.post("https://api.servidorinterno.com/puntuacion", json={"usuario_id": usuario_id, "puntos": puntos})
```