Evaluación

Ajedrez

Tiempo estimado	Dificultad	Tipo de actividad
20 horas	**Media**	**Evaluación**

Unión Europea
Fondo Europeo de Desarrollo Regional
Una manera de hacer Europa

GENERALITAT VALENCIANA
Conselleria d'Educació, Cultura i Esport

IES Mare Nostrum

Introducción

Se trata de hacer el juego del ajedrez. Para ello, un usuario podrá jugar contra otro.

Especificaciones

Para esta simulación, crearás **como mínimo** las siguientes clases:

- IChessBoard
  - Es una interfaz
  - Representa a un tablero, pero **sólo puede tener funciones de consulta**
  - Tiene como mínimo lo siguiente:
    - Obtener cuántas figuras tiene
    - Obtener una figura dado un índice
    - Obtener una figura dada una posición del tablero
    - Obtener el color del jugador al que le toca mover
    - Obtener el ganador/a
    - Saber si la partida ha terminado o no.
- IBoard
  - Es una interfaz
  - Hereda de IChessBoard
  - Tiene como mínimo lo siguiente:
    - Iniciar un tablero con todas las figuras
    - Borrar todo el tablero
    - Mover una figura de una posición a otra
- IFigure
  - Es una interfaz
  - Representa a cada figura que hay en un tablero
  - Tiene como mínimo y de manera obligatoria los siguientes métodos o propiedades
    - Obtener las coordenadas de la figura
    - Obtener el color de la figura
    - Obtener el tipo de figura que es (caballo, rey, reina, …) Esto lo deben implementar **obligatoriamente** las subclases finales.
    - Obtener una lista o un array con todas las posibles posiciones a las que se puede mover.

Restricciones

Tendrás que implementar las clases necesarias para hacer funcionar el juego siguiendo las especificaciones de las interfaces.

Se ha de seguir los siguientes puntos de forma obligatoria

- Bajo ningún concepto una **figura** puede tener acceso ni a un **IBoard**, ni a ninguna clase que lo implemente. Sí que puede tener acceso a un IChessBoard

Dificultades

Hay ciertos aspectos del juego que son difíciles de manejar, algunos con poca dificultad, y otros que incluso te obligan a romper el diseño. Entre otros se destacan los siguientes:

- Es posible que necesites tener información de si una figura se ha movido al menos una vez o no.
- Tendrás que ver cuando una partida termina o no.
- Puede que la partida termine por tablas.
- En la acción de mover una figura se han de tener en cuenta muchas cosas para saber si se puede efectuar el movimiento o no.
- El enroque es difícil de manejar puesto que involucra el movimiento de más de una figura.
- Puedes hacer que cada vez que un peón promocione, promocione directamente a una reina. En una aplicación real se debería preguntar al usuario, pero aquí no es necesario.
- Si el Rey está en jaque, los movimientos que se deberían permitir son todos aquellos que hagan que el rey no siga en jaque.
- Quizás este es el más difícil de todos, pero se debería contemplar que si un usuario mueve una ficha, y por culpa de haberla movido el rey entra en jaque mate, esa ficha no debería permitir que se mueva haciendo que no devuelva ninguna posición disponible en sus movimientos disponibles.

Ampliaciones

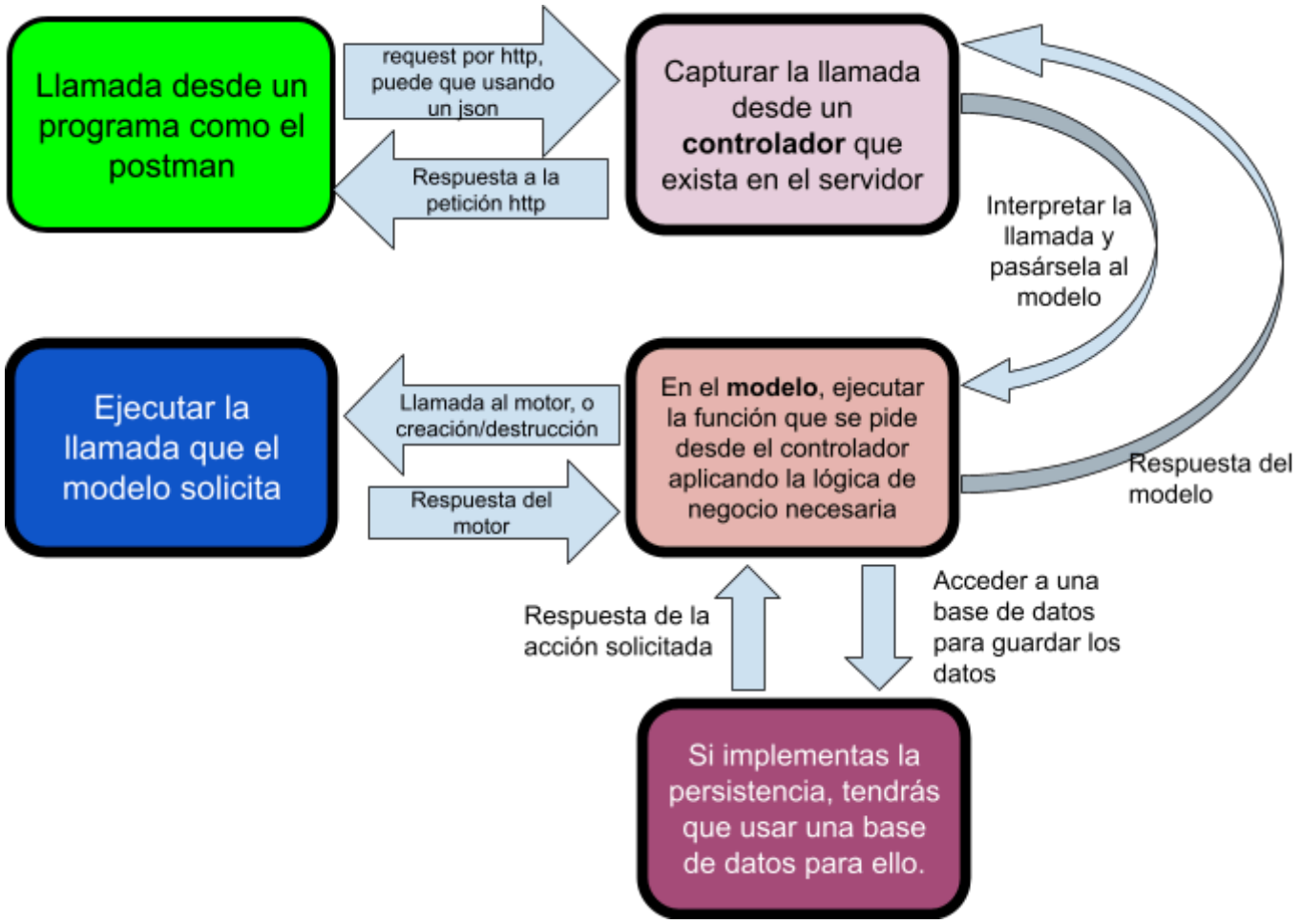
Desarrollar un API restful

Puedes implementar un sistema restful para contemplar llamadas http para gestionar el motor del juego.

Para hacer esto puedes usar el **MVC**. Recuerda que:

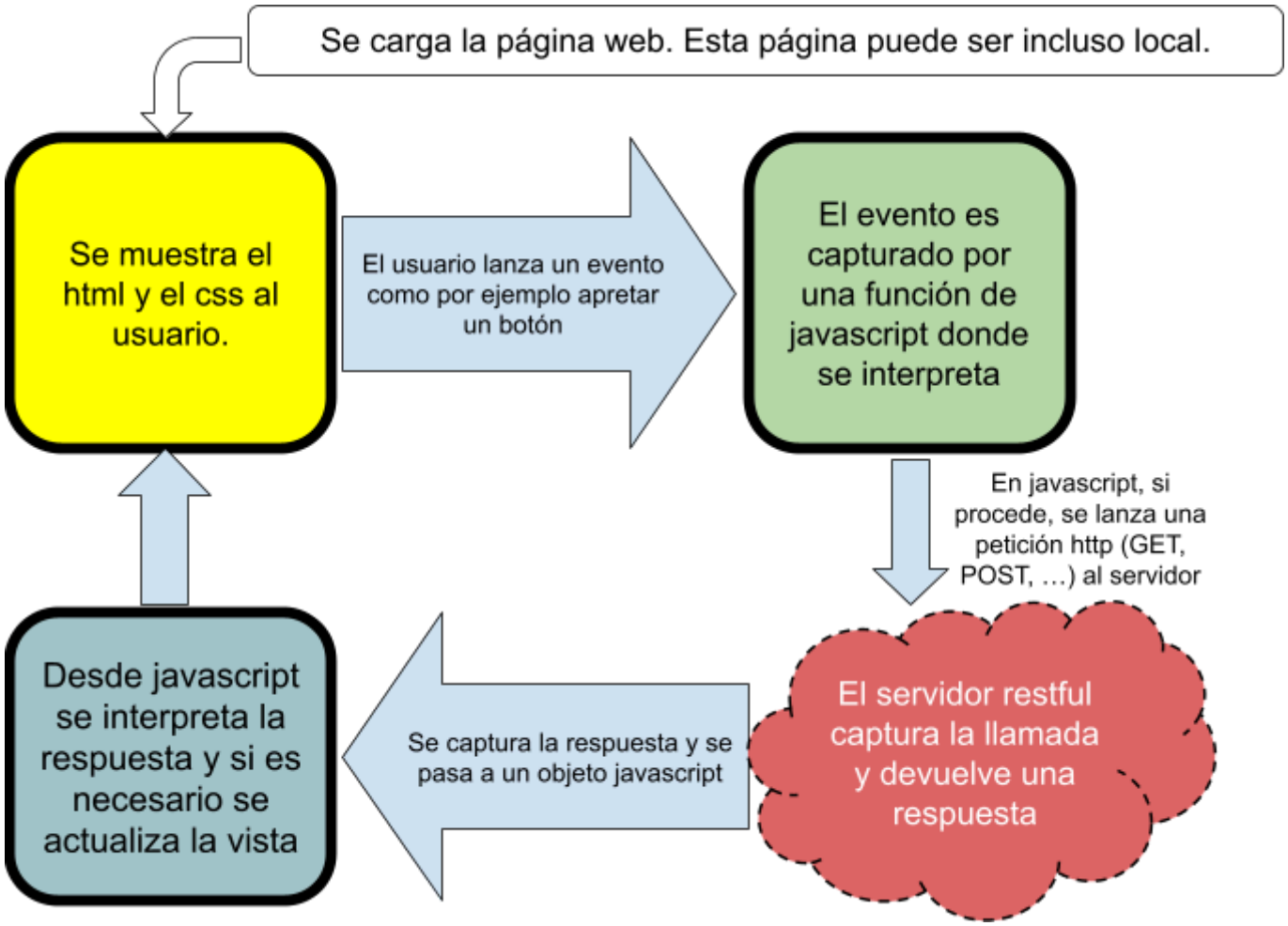
- La **vista** es la parte con la que interacciona el usuario, aunque para hacer el servidor con postman, esta parte no existe, o si quieres, puedes estimar que es el propio postman el encargado de interactuar con el usuario.
- El **modelo** es el que debe gestionar los datos y el estado de la aplicación. Estos datos pueden ser tanto persistentes (usar una base de datos) como no (usar la ram). También es el encargado principal de manejar la lógica de negocio de la aplicación, es aquí donde está el grueso de toda la lógica de negocio, aunque puede estar presente en más puntos.
- El **controlador** es un mero intermediario entre el modelo y la vista.

Para ello tienes que tener en cuenta lo siguiente:



Desarrollar un cliente visual http

Como prerrequisito para esta parte, debes hacer el apartado anterior del restful. Esta parte se programa en html, css y javascript. El cómo interacciona esta parte es la siguiente:



Hacer que el sistema sea persistente

Opcionalmente puedes querer mantener la persistencia entre ejecuciones, pero quedas advertido de que esta parte no es tan gratuita como parece.

Para ello tienes que tener en cuenta lo siguiente:

- Tienes que usar un SGBD como por ejemplo postgres, aunque para más sencillez, puedes guardar la base de datos en un archivo local usando SQLite.
- Si usas una base de datos puede ser que la parte del motor quede en un segundo plano. Si fuera un juego de tiempo real, evidentemente esto no sería así, pero dado que no lo es deberás tomar una de estas dos decisiones:
    - No usar una caché del motor. Esto es, cada vez que al modelo le llegue una petición (o al menos alguna de las veces), tendrá que
        - Consultar la base de datos
        - Con los datos recogidos de la base de datos, tendrá que recrear una instancia del motor
        - Se ha de preparar una respuesta accediendo al motor que se ha creado de manera efímera.
        - Destruir los datos del motor
    - Usar una caché del motor. La diferencia con el punto anterior es que esto no crea y borra el motor en cada llamada, sino que lo mantiene en memoria ram todo el tiempo. Esto es peligroso y difícil de mantener, puesto que puede haber inconsistencias entre lo que hay en ram y la base de datos

Calificación

- El motor del ajedrez se calificará como máximo con un **10**. Como mínimo hay que sacar un 5 para ser evaluado positivamente.
- (Optativo) La parte de Restful se calificará con un **+5** como máximo.
- (Optativo) La parte de cliente web, podrá aportar **+5** puntos, y debe tener entre otros
    - pseudo login del usuario
    - Ver las partidas en las que estoy jugando
    - Crear partidas
    - Ver una partida en concreto
    - Poder mover las figuras
- (Optativo) La parte de bases de datos, se puntuará con **+5** puntos como máximo.

Todos los apartados optativos han de funcionar al menos decentemente (aunque tengan fallos) para poder ser evaluados.

Para la interfaz web, se ha de demostrar que se entiende perfectamente lo que se esté haciendo aunque la haya generado Chat GPT.